PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-319595

(43)Date of publication of application : 12.12.1997

(51)Int.Cl. G06F 9/46

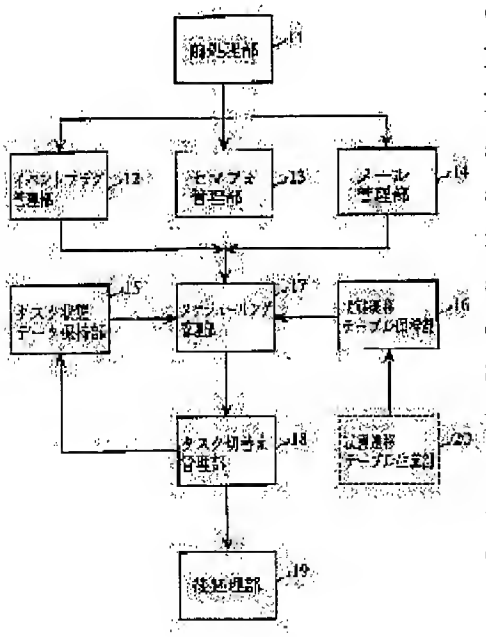
(21)Application number : 08-131484

(22)Date of filing : 27.05.1996

(71)Applicant : MATSUSHITA ELECTRIC IND CO LTD

(72)Inventor : IWAMURA YOSHIYUKI
SUMI FUMIO
NISHIHATA MOTOHIDE

(54) MULTITASK CONTROLLER



(57)Abstract:

PROBLEM TO BE SOLVED: To facilitate high-speed scheduling processing by referring to a state transition table when the generation of any event is detected, deciding the task to be next executed from a system state at the point of a time and switching tasks corresponding to the decision.

SOLUTION: A preprocessing part 11 performs preprocessing and an event flag managing part 12 manages the change of the system state accompanied with the generation of an event concerning an event flag. A semaphore managing part 13 manages the change of the system state accompanied with the generation of an event concerning the control of resources. A mail managing part manages the generation of an event concerning the mail and the change of the system state accompanied with the generation of that event. A task state data holding part 15 holds

an identifier showing the task under executing and a state transition table holding part 16 holds a state transition table for discriminating the system state after state transition from the current system state and the generated event. A scheduling managing part 17 discriminates the necessity of task switching and when switching is required, it is instructed to a switching managing part 18.

CLAIMS

[Claim(s)]

[Claim 1] In the multitasking control unit which changes the task which prepares for the system which performs two or more tasks, and said system performs according to generating of an event For every system state showing in which condition in various kinds of conditions including a running state, ready condition, and a standby condition of two or more of said tasks it is, respectively The state-transition table which holds the information which determines the task which should be performed next when each event occurs, A judgment means to judge the task which should refer to said state-transition table and should be performed next from the system state at the time when generating of an event is detected by event detection means to detect generating of an event, and said event detection means, The multitasking control unit characterized by having a task switching means to change a task according to the judgment of said judgment means.

[Claim 2] Said judgment means is a multitasking control unit according to claim 1 characterized by having an attaching part holding the system state at the time.

[Claim 3] Said state-transition table is a multitasking control unit according to claim 1 or 2 characterized by holding the system state of a transition place when each event occurs for every system state about the system state which may exist in said system.

[Claim 4] It is the multitasking control unit according to claim 3 which a state number is assigned by each system state in ascending order or descending order, and is characterized by said judgment means judging the task which should be performed next from the state number of the system state of the transition place currently held at the state-transition table in said state-transition table at it from the system state whose task with a high priority is a running state.

[Claim 5] Said multitasking control unit is a multitasking control unit given in claim 1 thru/or any of 4 they are. [which is characterized by having a state-transition table generation means to generate said state-transition table, from the system definition

table which includes further the definition with a task, the priority for every task, and the event that rises each task to ready condition]

[Claim 6] A condition generation means by which said state-transition table generation means generates all the combination of each task status from said system definition table, By deleting the combination showing the system state which cannot exist in said system from the combination which said condition generation means generated according to the regulation about a predetermined system state The multitasking control unit according to claim 5 characterized by having a condition extract means to extract the combination showing the system state which may exist in said system, and a table generation means to generate said state-transition table from the combination extracted by said condition extract means.

[Claim 7] Said condition extract means is a multitasking control unit according to claim 6 characterized by extracting the combination which expresses said system state according to the regulation about the system state based on the function of said system, and the regulation about the system state based on the reason of an application proper.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Field of the Invention] This invention relates to the multitasking control unit which performs multitasking control by the O ** rating system (it is called "OS" for short below).

[0002]

[Description of the Prior Art] In recent years, in fields, such as home electronics which perform microcomputer control, the multitasking control unit which changes a task using a synchronous transmission function is widely used on the occasion of the multitasking program execution which has two or more tasks. Here, in order to advance processing of two or more tasks synchronizing with a synchronous transmission function, it is the thing of the function which communicates between tasks. Hereafter, the conventional multitasking control unit using a synchronous transmission function is explained.

[0003] Drawing 23 is the block diagram showing the configuration of the conventional multitasking control device. The multitasking control device shown in this drawing has the pretreatment section 91, the event flag Management Department 92, the semaphore

Management Department 93, the mail administration section 94, the state-transition queue 95, the scheduling Management Department 96, the task switching Management Department 97, and the after-treatment section 98. The pretreatment section 91 performs processings with the need of carrying out as pretreatment each time, such as preservation processing of an identifier in which a multitasking control device expresses a calling agency immediately after driving with a system call instruction from a task or an interruption routine.

[0004] Here, the need of saving the identifier showing a calling agency originates in the contents of processing at the time of a return having a difference by the calling agency. That is, when a multitasking control unit drives from a task, even if it changes a task at the time of a return, a problem does not occur at all. However, when it drives from an interruption routine, and all interruption processings are ended and control returns to a task, a task may be changed, but like [at the time of interrupting where multiplex interruption is performed and driving from a routine, when other], the return point also interrupts, and when it is a routine, a task cannot be changed. This is based on the function of OS of it becoming impossible to return to the interruption routine of the call origin which is the original return point, when the return point interrupts, it is a routine and a task is changed.

[0005] Therefore, at the time of the return from a multitasking control unit, since it is necessary to identify a calling agency and to perform processing according to a calling agency, preservation processing of the identifier showing a calling agency is needed. The event flag Management Department 92 performs setting processing of an event flag etc., when a multitasking control device drives with a setting instruction or waiting instruction of an event flag from a task or an interruption routine.

[0006] Here, an event flag means the bit pattern currently held for two or more flags ID of every in the data area (it is called "OS field" below) which OS manages. The setting instruction of an event flag is an instruction which specifies Flag ID and a bit pattern from a task or an interruption routine, and sets up an event flag. Specifically, it sets to a task or an interruption routine. "setflag (setpattern, flagID)" Issue of an instruction which is said sets up the bit pattern specified as OS field which has the flag ID specified by the event flag Management Department 92 by flagID by setpattern. A setting instruction may be published from both a task and an interruption routine.

[0007] On the other hand, the waiting instruction of an event flag is an instruction which asks whether Flag ID and a bit pattern are specified from a task, and the bit pattern set as OS field which has the flag ID concerned, and the bit pattern specified by the task are in agreement. Specifically, it sets to a task. "waitflag (waitpattern, flagID)"

If an instruction which is said is published, the event flag Management Department 92 compares the bit pattern set as OS field which has ID specified by flagID with the bit pattern specified by waitpattern. About the task which published the instruction when both were in agreement as a result of the comparison, although it is possible to perform processing as it is, if not in agreement, it will be in a standby condition (it is called a "Wait condition" below). A Wait condition is in the condition of having interrupted activation of processing until a task will be in ready condition (it is called a "Ready condition" below) according to generating of an event (it calls "rising" hereafter that the task of a Wait condition will be in a Ready condition according to generating of an event). If it is an example in this case, a Wait condition will continue with the setting instruction of an event flag published from other tasks or interruption routines until both bit pattern is in agreement. In addition, the waiting instruction of an event flag is published only from a task, and is not published from an interruption routine.

[0008] In a Wait condition, if a task rises according to generating of an event, the task concerned will be in a Ready condition first, and when the priority of the task concerned is the highest, or when the task of a Ready condition does not exist in others, if the situation which can perform the task concerned is ready, it will be in a running state (it is called a "Run condition" below). In the waiting instruction of an event flag, the bit pattern specified by waitpattern is held with assignment of Flag ID at TCB (Task Control Block) assigned to each task.

[0009] The event flag Management Department 92 searches whether the task which specifies a bit pattern to the flag which has the flag ID concerned, and is in the Wait condition exists after setting processing of an event flag, when a multitasking control device drives with the setting instruction of an event flag. When the bit pattern which the task of a Wait condition exists as a result of retrieval, and is specified by the task of a Wait condition, and the bit pattern of OS field after setting processing are in agreement, the task of a Wait condition rises. In this case, the event flag Management Department 92 notifies the scheduling Management Department 96 of rising of the task by generating of an event.

[0010] Moreover, when a multitasking control device drives with the waiting instruction of an event flag, the event flag Management Department 92 compares the bit pattern and the bit pattern of OS field which were specified by the task as above-mentioned, and if in agreement, it will continue activation of processing of a task as it is. Since a task will be in a Wait condition when not in agreement, the event flag Management Department 92 notifies the scheduling Management Department 96 of change in the Wait condition of a task.

[0011] The semaphore Management Department 93 has the counter which was assigned by the resource and which holds the amount of the resource in the time which can be assigned for every ID, and when a multitasking control device drives by a resource acquisition demand and resource deallocation demand from a task or an interruption routine, it performs an update process of a counter etc. Here, a resource acquisition demand is an instruction which specifies the acquisition amount required of ID of a resource, and a resource, and is published from a task. concrete -- for example, -- "waitsem (semID, reqNo)" ** -- if an instruction which is said is published, the semaphore Management Department 93 measures the amount required specified by reqNo, and the amount which is held at the counter about the resource which has ID specified by semID and which can be assigned. Although the semaphore Management Department 93 will subtract the acquisition amount required from the value of a counter, the demanded resource will be assigned to a task and continuation activation of the processing of the task concerned will be carried out if the amount which can be assigned is more than the acquisition amount required When the amount which can be assigned is less than the acquisition amount required, the task concerned will be in a Wait condition until the amount of the resource concerned which can be assigned increases by the resource deallocation demand from other tasks and it becomes more than the acquisition amount required.

[0012] On the other hand, a resource deallocation demand is an instruction which specifies the amount of release requests of ID of a resource, and a resource, and is published from a task. concrete -- for example, -- "sigsem (semID, relNo)" ** -- issue of an instruction which is said adds the value of the counter about the resource which has ID specified by semID. In addition, although a resource deallocation demand may be published from both a task and an interruption routine, a resource acquisition demand is published only from a task and published from an interruption routine.

[0013] The semaphore Management Department 93 searches whether the task which waits for the increment in the amount of the resource concerned which can be assigned, and is in the Wait condition exists after addition of the value of a counter, when a multitasking control unit drives by resource deallocation demand. When the amount of resources which the task of a Wait condition exists as a result of retrieval, and is demanded by the task of a Wait condition is below the value of the counter after addition, the task of a Wait condition rises. In this case, the semaphore Management Department 93 notifies the scheduling Management Department 96 of rising of the task by generating of an event.

[0014] If the amount of the resource currently held at the counter as above-mentioned

which can be assigned is more than the acquisition amount required when a multitasking control unit drives by resource acquisition demand, the semaphore Management Department 93 will subtract the value of a counter, and will continue activation of processing of a task as it is. Since the task concerned will be in a Wait condition when the amount which can be assigned is less than the acquisition amount required, the semaphore Management Department 93 notifies the scheduling Management Department 96 of change in the Wait condition of a task.

[0015] The mail administration section 94 performs management processing of e-mail etc., when a multitasking control device drives by the RECEIVE statement or SEND statement of e-mail from a task or an interruption routine. E-mail is held with a message at the transmitted order for two or more mail boxes ID of every. The RECEIVE statement of e-mail is an instruction which asks whether a mail box ID is specified from a task and e-mail is held at the mail box concerned. concrete -- for example, -- "waitmsg (boxID, ptr)" ** -- issue of an instruction which is said searches whether e-mail is held at the mail box specified by boxID. If e-mail is held, the task which published the RECEIVE statement can receive the mail, and can continue activation of processing. Since the address of the location where a message exists is returned to ptr, a task continues processing using the data held in the message region.

[0016] If e-mail is not held, a task will be in a Wait condition until e-mail is transmitted to the mail box concerned from other tasks. Moreover, the SEND statement of e-mail is an instruction which specifies a mail box ID and the address of a message maintenance field, and transmits e-mail from a task or an interruption routine. On a concrete target "sendmsg (boxID, ptr)" Issue of an instruction which is said transmits the mail accompanied by the message held to the address specified as the mail box specified by boxID by ptr.

[0017] In addition, although the SEND statement of e-mail may be published from both a task and an interruption routine, the RECEIVE statement of e-mail is published only from a task, and is not published from an interruption routine. When a multitasking control unit drives by the SEND statement of e-mail, the mail administration section 94 searches whether the task which waits for transmission of the mail to the mail box concerned, and is in the Wait condition exists, after performing transmitting processing of e-mail.

[0018] If the task of a Wait condition exists as a result of retrieval, since the task of a Wait condition will rise, the mail administration section 94 notifies the scheduling Management Department 96 of rising of the task by generating of an event. If e-mail is held at the mail box specified as above-mentioned when a multitasking control unit

drives by the RECEIVE statement of e-mail, the mail administration section 94 will pass e-mail to a task, and will continue activation of processing of a task as it is. Since the task which published the RECEIVE statement will be in a Wait condition when e-mail is not held, the mail administration section 94 notifies the scheduling Management Department 96 of change in the Wait condition of a task.

[0019] The state-transition queue 95 consists of a Run queue and a Ready queue. A Run queue is a field where TCB of the task of a Run condition is connected, and a Ready queue is a field in which the task of a Ready condition forms the queue in order of priority. The scheduling Management Department 96 receives the notice from the event flag Management Department 92, the semaphore Management Department 93, and the mail administration section 94, and it performs scheduling so that it may connect with a Run queue sequentially from a task with a high priority, while it manages the order of a list of the queue of a Ready queue, separating the task which changed into the Wait condition from the Run condition from a Run queue, or measuring the priority of each task of a Ready condition about the task which rose according to generating of an event.

[0020] The task switching Management Department 97 actually performs the spawn process of tasks, such as evacuation/return of a register, according to the scheduling of the scheduling Management Department 96. As for the after-treatment section 98, renewal of the identifier of the call origin saved in the pretreatment section 91 etc., deletion, etc. perform processing which has the need of carrying out each time at the last of multitasking control. About the multitasking control unit constituted as mentioned above, the actuation is hereafter explained using an example. Drawing 24 is drawing showing the condition of the state-transition queue 95 in this example.

[0021] For example, three tasks A, B, and C exist, for Task A, the highest priority and Task B are [the following priority and Task C] the minimum priorities, and Task A assumes the case where a Run condition and Task C are in a Ready condition for a Wait condition and Task B. It explains actuation when the SEND statement of the mail to a mail box X is published from the task B which is in a Run condition that Task A is in the Wait condition of waiting to transmit e-mail to a mail box X.

[0022] The pretreatment section 91 performs OS common processing of preservation of the identifier showing having driven by the SEND statement of e-mail from Task B etc. first. In this example, since it is called by the SEND statement of e-mail, processing by the mail administration section 94 is performed. The mail administration section 94 searches whether the task which waits for transmission of the mail to a mail box X, and is in the Wait condition exists, after transmitting e-mail to a mail box X. Since it becomes clear that Task A rises and it will be in a Ready condition as a result of

retrieval, the mail administration section 94 notifies the scheduling Management Department 96 of rising of the task by transmission of e-mail.

[0023] The scheduling Management Department 96 receives the notice of the mail administration section 94, and changes the condition of the state-transition queue 95 like drawing 24 (a) to drawing 24 (b) first. That is, since the task A which was in the Wait condition will be in a Ready condition, TCB of Task A is connected with a Ready queue. Under the present circumstances, since the priority of Task A is higher than the priority of Task C, it will be tied to a Ready queue in order of Task A and Task C as a result of the search of a Ready queue.

[0024] Next, the scheduling Management Department 96 measures the priority of the task B which is in a Run condition, and the task A which is a task with the highest priority in the task which is in a Ready condition, and changes the condition of the state-transition queue 95 like drawing 24 (b) to drawing 24 (c). That is, since the priority of Task A is higher than the priority of Task B, TCB of Task A is connected with a Run queue, and TCB of Task B is connected with a Ready queue. In this case, since the priority of Task B is higher than the priority of Task C, it is connected with a Ready queue in order of Task B and Task C as a result of the search of a Ready queue.

[0025] The scheduling Management Department 96 detects change of the task connected with the Run queue, and notifies the task switching Management Department 97 of being changed [of a task]. The task switching Management Department 97 performs processing required for actual task switching, such as evacuation/return of a register, and hands control to the after-treatment section 98. After the after-treatment section 98 performs OS common processing of deletion of an identifier etc. which shows having driven from Task B, processing of a multitasking control unit is completed. Processing of Task A is performed after processing termination.

[0026]

[Problem(s) to be Solved by the Invention] However, it sets to the above conventional multitasking control units. When the Wait condition of a task is canceled by generating of events, such as a setup of an event flag, and transmission of e-mail, and it changes into a Ready condition, a queue search, queue connection, etc. need to be queue operated, And comparison processing of the priority of the task which changed into the Ready condition, and the task of a Run condition, Comparison processing of the priority of the tasks of a Ready condition, the Run queue at the time of task switching, and a Ready queue need to be operated etc., and it had the trouble that scheduling processing was slow. When it was necessary to carry out as a result, for example, high-speed control

processing, the problem of being unable to use the general-purpose multitasking OS which has high efficiency had arisen.

[0027] This invention aims at offering the multitasking control unit which enables high-speed scheduling processing in view of the above-mentioned trouble.

[0028]

[Means for Solving the Problem] The multitasking control unit concerning this invention in order to solve the above-mentioned trouble In the multitasking control unit which changes the task which prepares for the system which performs two or more tasks, and said system performs according to generating of an event For every system state showing in which condition in various kinds of conditions including a running state, ready condition, and a standby condition of two or more of said tasks it is, respectively The state-transition table which holds the information which determines the task which should be performed next when each event occurs, A judgment means to judge the task which should refer to said state-transition table and should be performed next from the system state at the time when generating of an event is detected by event detection means to detect generating of an event, and said event detection means, It has a task switching means to change a task according to the judgment of said judgment means.

[0029] Moreover, said judgment means can also have an attaching part holding the system state at the time. Moreover, said state-transition table can hold the system state of a transition place when each event occurs for every system state about the system state which may exist in said system.

[0030] Furthermore, in said state-transition table, to each system state, a state number is assigned in the system state whose task with a high priority is a running state to ascending order or descending order, and said judgment means can judge the task which should be performed next from the state number of the system state of the transition place currently held at the state-transition table. Said multitasking control unit can also be equipped with a state-transition table generation means to generate said state-transition table, from the system definition table which includes further the definition with a task, the priority for every task, and the event that rises each task to ready condition.

[0031] Moreover, a condition generation means by which said state-transition table generation means generates all the combination of each task status from said system definition table, By deleting the combination showing the system state which cannot exist in said system from the combination which said condition generation means generated according to the regulation about a predetermined system state It is also

possible to have a condition extract means to extract the combination showing the system state which may exist in said system, and a table generation means to generate said state-transition table from the combination extracted by said condition extract means.

[0032] Furthermore, said condition extract means can extract the combination which expresses said system state according to the regulation about the system state based on the function of said system, and the regulation about the system state based on the reason of an application proper.

[0033]

[Embodiment of the Invention] Hereafter, it explains, referring to a drawing about the gestalt of 1 operation of this invention. Drawing 1 is the block diagram showing the configuration of the multitasking control device concerning the gestalt of this operation. The multitasking control device shown in this drawing has the pretreatment section 11, the event flag Management Department 12, the semaphore Management Department 13, the mail administration section 14, the task state data-hold section 15, the state-transition table attaching part 16, the scheduling Management Department 17, the task switching Management Department 18, and the after-treatment section 19.

[0034] In addition, the multitasking control unit of the gestalt of this operation is the same as the multitasking control unit explained by the Prior art in the point realized when CPU performs OS, and CPU about the point which is what is driven whenever the instruction of the setting instruction of an event flag and a waiting instruction, the acquisition demand of a resource and a release request, the SEND statement of e-mail, a RECEIVE statement, etc. is executed.

[0035] The pretreatment section 11 performs processings with the need of carrying out as pretreatment each time, such as preservation processing of an identifier in which the multitasking control device of the gestalt of this operation expresses a calling agency immediately after driving from a task or an interruption routine. The event flag Management Department 12 manages change of the system state accompanying generating of the event about an event flag, and generating of the event concerned, when a multitasking control device drives with a setting instruction or waiting instruction of an event flag from a task or an interruption routine. A system state is information which shows whether each task is in which condition of a Run condition, a Ready condition, and a Wait condition.

[0036] When setting processing of an event flag is specifically performed when a multitasking control device drives with the setting instruction of an event flag, and also a system state has change by setup of the flag concerned, the scheduling Management

Department 17 is notified of generating of the event of a setup of the flag which has the flag ID specified with a setting instruction. Moreover, when the task driven when a multitasking control device drove with the waiting instruction of an event flag changes to a Wait condition, the scheduling Management Department 17 is notified of generating of the event of change in the Wait condition of a task of having driven the multitasking control device.

[0037] The semaphore Management Department 13 manages change of the system state accompanying generating of the event about control of a resource, and generating of the event concerned, when a multitasking control device drives by an acquisition demand or resource deallocation demand of a resource from a task or an interruption routine. When addition processing of a counter is specifically performed when a multitasking control unit drives by the release request of a resource, and also the resource concerned has change by disconnection at a system state, the scheduling Management Department 17 is notified of generating of the event of disconnection of a resource. Moreover, when the task driven when a multitasking control unit drove by the acquisition demand of a resource changes to a Wait condition, the scheduling Management Department 17 is notified of generating of the event of change in the Wait condition of a task of having driven the multitasking control unit.

[0038] The mail administration section 14 manages change of the system state accompanying generating of the event about e-mail, and generating of the event concerned, when a multitasking control device drives by the RECEIVE statement or SEND statement of e-mail from a task or an interruption routine. When a multitasking control unit drives by the SEND statement of e-mail and a system state specifically has change by transmission of the mail concerned, the scheduling Management Department 17 is notified of generating of the event of transmission of the mail to the mail box which has the mail box ID specified by the SEND statement. Moreover, when the task driven when a multitasking control unit drove by the RECEIVE statement of e-mail changes to a Wait condition, the scheduling Management Department 17 is notified of generating of the event of change in the Wait condition of a task of having driven the multitasking control unit.

[0039] The identifier which shows the task under activation is held at the task state data-hold section 15. The state-transition table attaching part 16 holds the state-transition table for judging a current system state and the system state after [the generated event to] a state transition. A state-transition table is beforehand generated in the state-transition table generation section 20. About the DS and the generation method of a state-transition table, it mentions later.

[0040] The notice from the event flag Management Department 12, the semaphore Management Department 13, and the mail administration section 14 is received, and with reference to the contents of the task state data-hold section 15 and the state-transition table attaching part 16, the scheduling Management Department 17 judges whether it is the need, and task switching directs it to the task switching Management Department 18, when task switching is required. The task switching Management Department 18 updates the contents of the task state data-hold section 15 while performing the spawn process of tasks, such as evacuation/return of a register, according to directions of the scheduling Management Department 17.

[0041] As for the after-treatment section 19, renewal of the identifier of the call origin saved in the pretreatment section 11 etc., deletion, etc. perform processing which has the need of carrying out each time at the last of multitasking control. Drawing 2 is a flow chart which shows the contents of processing of the multitasking control device of the gestalt of this operation. The multitasking control device of the gestalt of this operation starts actuation in response to the system call instruction from a task or an interruption routine (S101).

[0042] First, the pretreatment section 11 performs OS common processing of saving the identifier of the task or interruption routine which performed the system call (S102). Next, based on the contents of the system call instruction, processing depended for any of the event flag Management Department 12, the semaphore Management Department 13, and the mail administration section 14 being is performed. For example, if the instruction at the time of a system call is a setting instruction of an event flag, flag setting instruction processing will be performed by the event flag Management Department 12 (S104). Moreover, if it is the waiting instruction of an event flag, waiting instruction processing for a flag will be performed by the event flag Management Department 12 (S105).

[0043] Moreover, if the instruction at the time of a system call is the release request of a resource, resource deallocation demand processing will be performed by the semaphore Management Department 13 (S106), and if it is a resource acquisition demand, resource acquisition demand processing processing will be performed by the semaphore Management Department 13 (S107). Moreover, if the instruction at the time of a system call is an e-mail SEND statement, e-mail transmitting processing will be performed by the mail administration section 14 (S108), and if it is the RECEIVE statement of e-mail, e-mail reception will be performed by the mail administration section 14 (S109).

[0044] Drawing 3 is a flow chart which shows the detailed contents of processing of flag setting instruction processing which the event flag Management Department 12

performs. The event flag Management Department 12 sets up the bit pattern specified as the event flag field of OS shown by the flag ID specified in the setting instruction of an event flag (S201), and searches the task which specifies the flag ID concerned and is in the Wait condition (S202). If the task which is in the Wait condition exists as a result of retrieval (S203:Yes), the bit pattern set up with a setting instruction will investigate whether it is in agreement with the bit pattern specified by the task of a Wait condition (S204). Since a system state will change when the task of a Wait condition rises if the bit pattern is in agreement, the event flag Management Department 12 clears a bit pattern (S205), and notifies the scheduling Management Department 17 of generating of the event of a setup of an event flag (S206). In addition, a specification, the contents of processing, etc. of the OS may not perform the clearance of the bit pattern in step S205.

[0045] the task which is in the Wait condition -- not existing (S203:No) -- since it means that there is no change in a system state in any way when a bit pattern is not in agreement (S204:No), even if a task exists, processing of the event flag Management Department 12 is ended as it is. Drawing 4 is a flow chart which shows the detailed contents of processing of waiting instruction processing for a flag which the event flag Management Department 12 performs. It compares whether event flag Management Department's 12 bit pattern currently held to OS field which has the flag ID specified in the waiting instruction of a flag corresponds with the bit pattern specified with a waiting instruction (S211). If in agreement, since it means that change does not occur in a system state at all, the event flag Management Department 12 clears a bit pattern (S214), and ends processing. When not performing clear processing of a bit pattern, a certain thing is also the same as that of flag setting instruction processing.

[0046] If not in agreement, since the task which published the waiting instruction will be in a Wait condition, the event flag Management Department 12 notifies the scheduling Management Department 17 of generating of the event of change in the Wait condition of a task (S213). Drawing 5 is a flow chart which shows the detailed contents of processing of the resource deallocation demand processing which the semaphore Management Department 13 performs. The semaphore Management Department 13 adds the amount of the resource opened wide to the counter holding the amount of the resource shown by ID specified in the resource deallocation instruction which can be assigned (S301), and searches whether the task which is in the Wait condition as acquisition waiting of the resource concerned exists (S302). As a result of retrieval, when the task which is in the Wait condition exists, it judges whether the task which rises by addition of the counter in step S301 occurs (S304). If the task which rises occurs, only the resource acquisition amount required of the task concerned which rises will

subtract a counter (S305), and will notify the scheduling Management Department 17 of generating of the event of disconnection of a resource (S306).

[0047] Since it means that there is no change in a system state in any way when having exceeded the value of the counter after the acquisition amount required of the resource by the task concerned updating (S304:No), even if the case (S303:No) where the task which is in the Wait condition does not exist, and the task of a Wait condition exists, processing of the semaphore Management Department 13 is ended as it is.

[0048] Drawing 6 is a flow chart which shows the detailed contents of processing of the resource acquisition demand processing which the semaphore Management Department 13 performs. The semaphore Management Department 13 compares the value currently held at the counter holding the amount of the resource shown by ID which the resource acquisition demand was ordering and was specified which can be assigned with the acquisition amount required of the resource which was ordering and was specified (S311). When the value currently held at the counter is more than the acquisition amount required of a resource, it means securing the resource which the task concerned required and continuing activation of processing as it is. Therefore, the semaphore Management Department 13 subtracts the amount required of a resource from the value of a counter (S314), and ends processing.

[0049] Since the task concerned will be in a Wait condition as resource waiting when the value currently held at the counter is less than the acquisition amount required of a resource, the semaphore Management Department 13 notifies the scheduling Management Department 17 of generating of the event of change in the Wait condition of a task (S313). Drawing 7 is a flow chart which shows the detailed contents of processing of the e-mail transmitting processing which the mail administration section 14 performs.

[0050] The mail administration section 14 transmits a message to the mail box shown by ID which was ordering and was specified (S401). Then, it searches whether the task which waits for transmission of the mail to the mail box concerned, and is in the Wait condition exists (S402). When the task of a Wait condition exists, it means that the task of the Wait condition concerned rises. In this case, the mail administration section 14 notifies the scheduling Management Department 17 of generating of the event of reception of delivery (S404) and mail for e-mail at the task of the Wait condition concerned (S405). If the task of a Wait condition does not exist (S403:No), since it means that there is no change in a system state in any way, processing of the mail administration section 14 is ended as it is.

[0051] Drawing 8 is a flow chart which shows the detailed contents of processing of the

e-mail reception which the mail administration section 14 performs. The mail administration section 14 searches the contents of the mail box which has the mail box ID which was ordering and was specified (S411). If e-mail exists in the specified mail box, since it means that the task concerned receives the mail and can carry out continuation activation of the processing as it is, the mail administration section 14 passes e-mail to the called task (S414), and ends processing as it is.

[0052] Since it will be in a Wait condition until the task which published the RECEIVE statement has transmission of the mail from other tasks if e-mail does not exist in the specified mail box (S412:No), the mail administration section 14 notifies the scheduling Management Department 17 of generating of the event of change in the Wait condition of a task (S413). As mentioned above, termination of the processing depended for any of the event flag Management Department 12, the semaphore Management Department 13, and the mail administration section 14 being looks at whether it returned to the flow chart of drawing 2 , and the notice of generating of an event to the scheduling Management Department 17 from each event Management Department was performed (S110). Since it means that that the notice of generating of an event is not performed here does not have change in a system state in any way, processing of the scheduling Management Department 17 and the task switching Management Department 18 is not performed, but it progresses to step S111 as it is (S110:No).

[0053] When the notice to the scheduling Management Department 17 from each event Management Department is, processing of the scheduling Management Department 17 is performed (S111). Drawing 9 is a flow chart which shows the detailed contents of processing of the scheduling Management Department 17. The scheduling Management Department 17 detects generating of an event in response to the notice from the event flag Management Department 12, the semaphore Management Department 13, or the mail administration section 14 first (S501), and refers to the state-transition table attaching part 16. In the gestalt of this operation, since the scheduling Management Department 17 holds the state number (the state number is assigned by the system state so that it may mention later) showing a current system state, it judges a current state number and the state number after [the generated event to] a state transition (S502).

[0054] Drawing 15 is drawing showing the DS of a state-transition table. Although the generation method of a state-transition table is explained to a detail later, the operation of a state-transition table is explained briefly here. The state-transition table shown in drawing 15 takes the present state number along an axis of ordinate, and takes the event generated on the axis of abscissa. A state number is a number assigned for every

system state. Moreover, when the task of a Run condition will be in a Wait condition as an axis of abscissa in this example (it is called "change in the Wait condition" below), When the setting instruction of a flag was published and a bit pattern is set as the event flag field whose flag ID is "FLAG_A" (it is called "a setup of Flag A" below), When e-mail is transmitted to the mail box whose mail box ID is "MBX_B" (it is called "transmission of Message B" below), In the system state which four events when e-mail is transmitted to the mail box whose mail box ID is "MBX_C" (it is called "transmission of Message C" below) are taken, and is shown with a current state number The state number after a state transition when each event occurs is shown.

[0055] That is, on the state-transition table of drawing 15 , when the task which drove the multitasking control unit in the system state of a state number 1 changes into a Wait condition, it means changing to the system state of a state number 4, for example. Here, the state number is assigned so that it may have as small a state number as the system state whose task with a high priority is in a Run condition. In the system state whose state numbers are 1, 2, and 3 in the example of drawing 15 , the task A with the highest priority is in a Run condition, in the system state whose state numbers are 4 and 5, the task B with a high priority is in a Run condition next, and the task C with the lowest priority is in a Run condition in the system state whose state number is 6. Thus, by assigning a state number, it becomes easy to distinguish the task which will be in a Run condition from a state number.

[0056] The scheduling Management Department 17 judges first the task which is in a current Run condition, i.e., the task which drove the multitasking control unit, with reference to the task state data currently held at the task state data-hold section 15. Next, task switching judges whether it is the need by what (S503) the task which will be in a Run condition is judged and both are compared for after a state transition from the state number showing the system state after the state transition obtained with reference to a state-transition table (S504).

[0057] Here, since a calling agency interrupts, and task switching is impossible in being a routine, it judges with task switching not being required (S505). By the above judgment, when task switching is required, the change of a task is required of the task switching Management Department 18, and processing (S506) is ended. In addition, in being a routine (S505:Yes), in case a calling agency interrupts, and it returns to an interruption routine, task switching cannot be performed, but when processing by the interruption routine is completed and control returns to a task, it is necessary to change a task. Therefore, processing called marking of task switching is performed (S507). Marking of task switching is processing which saves the identifier of the task which

should be performed after the state transition concerned etc., in order to perform task switching in next timing about the task after the state transition in step S504 (when control returns to a task). It carries out by the approach of specifically saving the identifier showing the task which should be changed to the predetermined field established in OS field.

[0058] After processing of the scheduling Management Department 17 is completed, it returns to the flow chart of drawing 2 , and when task switching is unnecessary (S112:No), it progresses to step S114 as it is, without performing processing by the task switching Management Department 18. When task switching is required, processing of evacuation/return of a register and an update process of the task state data-hold section 15 are performed by the task switching Management Department 18 (S113), common processing of the renewal of informational which expresses a calling agency with the after-treatment section 19, deletion, etc. is performed (S114), and processing of a multitasking control unit is completed.

[0059] As mentioned above, in the multitasking control unit of the gestalt of this operation, since the approach of judging the system state after a state transition by processing neither queue actuation nor the comparison of the priority between tasks, but referring to a state-transition table is taken when generating of an event is detected, it becomes possible to perform scheduling processing at a high speed. Next, the generation method of a state-transition table is explained.

[0060] Drawing 10 is the block diagram showing the detailed configuration of the state-transition table generation section 20. As shown in this drawing, the state-transition table generation section 20 consists of the condition generation section 21, the condition extract section 22, and the table generation section 23. In addition, the system definition table attaching part 30 shown in this drawing is a part holding a system definition table, and exists also in the conventional multitasking control unit. Most, about the contents of the table which the system definition table attaching part 30 holds, it may change with specifications of OS etc.

[0061] Drawing 11 is drawing showing an example of the contents of the system definition table. Various these drawings express a part required for generation of a state-transition table in the existing system definition table. As shown in this drawing, in a system definition table, the task definition table shown in drawing 11 (a) and the synchronous transmission functional definition table shown in drawing 11 (b) exist. The task definition table of drawing 11 (a) defines the priority of a task name and a task, stack size, etc. The priority of this drawing means that a priority is so high that the figure is small. Although the example of the gestalt of this operation defines three tasks,

Task A, Task B, and Task C, it is also possible for there to be especially no limit in the number of tasks, and to give the same priority to two or more tasks. The flag ID relevant to the event which may be generated, ID of a resource, and a mail box ID, the event concerned and a task name with relation are defined as the synchronous transmission functional definition table of drawing 11 (b). Although the definition of the purport to which the event flag whose flag ID is "FLAG_A" is related with Task A is made in the example of this drawing, a setup of the event flag which has the flag ID "FLAG_A" is detected as generating of an event, and this means that Task A rises according to generating of the event concerned.

[0062] A mail box ID "MBX_B" "the same is said of MBX_C], transmission of the mail to the mail box concerned is detected as generating of an event, and it means that Task B and Task C rise according to generating of the event concerned, respectively. The condition generation section 21 generates a system state table with reference to the system definition table currently held at the system definition table attaching part 30.

[0063] Drawing 12 is drawing showing the DS of a system state table. As shown in this drawing, a system state table defines all the combination in the condition that the task defined as the system definition table can take. Although three tasks are defined by the example of this drawing and each priorities of all differ, it is possible also when the priority of two or more tasks is the same. In such a case, about the task of the same priority, it is regarded as that from which a system state differs by the sequence which rose, and a system state table is generated.

[0064] The condition extract section 22 contains the condition extract section 221 based on OS regulation, and the condition extract section 222 based on an application proper regulation. For example When the regulation (it is called an "application proper regulation" below) about the system state based on the reason of an application proper as shown in the regulation (it is called "OS regulation" below) and drawing 13 (b) about the system state based on a function of OS as shown in drawing 13 (a) exists The condition extract section 221 based on OS regulation extracts only the system state corresponding to OS regulation from a system state table, and the condition extract section 222 based on an application proper regulation extracts further only the system state corresponding to an application proper regulation. The condition extract section 22 generates the system state minimum table by reassigning a state number to the extracted system state. Drawing 14 (b) is an example of the system state minimum table.

[0065] The table generation section 23 generates a state-transition table from the system state minimum table. Next, the contents of processing of the state-transition

table generation section 20 are explained. Drawing 16 is a flow chart which shows the contents of processing of the state-transition table generation section 20. The state-transition table generation section 20 performs processing by the condition generation section 21 first (S701). The condition generation section 21 generates the system state table shown in drawing 12 with reference to the system definition table shown in drawing 11 as above-mentioned. A system state table is generated by specifically extracting a task name from the task definition table shown in drawing 11 (a), extracting the event related from the synchronous transmission functional definition table shown in drawing 11 (b), and defining all the combination in the condition that each task can take. Here, Wait (Fa) is in the Wait condition of waiting for a setup of the event flag which has Flag ID "FLAG_A", and means that Wait (Mb) is in the Wait condition of waiting for transmission of the mail to the mail box which has a mail box ID "MBX_B." It is the same as that of Wait (Mb) also about Wait (Mc).

[0066] After ending generation of a system state table, it returns to the flow chart of drawing 16 , and the state-transition table generation section 20 performs processing by the condition extract section 221 based on OS regulation first as processing by the condition extract section 22 (S702). Drawing 17 is a flow chart which shows the detailed contents of processing of the condition extract processing based on OS regulation. The condition extract section 221 based on OS regulation deletes the combination in which the task of a Run condition exists by two or more systems from a system state table based on OS regulation (1) first shown in drawing 13 (a) (S801). The combination showing the system state whose state numbers are 5, 6, 8, 9, 11-18, and 20-27 is extracted from a system state table by this processing.

[0067] Next, the condition extract section 221 based on OS regulation deletes the combination from which the task of the highest priority is in the Ready condition based on OS regulation (2) (S802). In the example of the gestalt of this operation, since the task of the highest priority is Task A, the combination which expresses the system state whose state numbers are 5, 6, 8, 9, 20-27 by this processing is extracted.

[0068] Next, the condition extract section 221 based on OS regulation deletes the combination from which the task with a priority higher than the task is in the Ready condition, when the task of a Run condition exists based on OS regulation (3) (S803). In the example of the gestalt of this operation, since the priority is high in order of Task A, Task B, and Task C, if the result of the extract in step S802 is taken into consideration, the combination whose task B is in a Ready condition and whose task C is in a Run condition will be set as the object of deletion by processing of this step. That is, the combination which expresses the system state whose state numbers are 5, 6, 8, 9, 20, 21,

23-27 by this processing is extracted.

[0069] At the last, the combination in which, as for the condition extract section 221 based on OS regulation, the task of a Run condition does not exist by the task of a Ready condition existing based on OS regulation (4) is deleted (S804). The combination which expresses the system state whose state numbers are 5, 6, 8, 9, 20, 21, 25, and 27 by this processing is extracted. After ending the condition extract processing based on OS regulation, it returns to the flow chart of drawing 16 , and the condition extract section 22 performs condition extract processing based on an application proper regulation (S703). The condition extract section 222 based on an application proper regulation deletes the combination from which Task B and Task C will be in a Wait condition at coincidence based on the application proper regulation shown in drawing 13 (b) from the system state extracted by the condition extract processing based on OS regulation. The combination which expresses the system state ** Li and whose state number are 5, 6, 8, 20, 21, and 25 to this processing is extracted.

[0070] By the above extract processing, the condition extract section 22 obtains the system state table after a condition extract as shown in drawing 14 (a). In respect of the size effectiveness of a table etc., the condition extract section 22 reassigns a state number so that advantageously, and it generates the system state minimum table as finally shown in drawing 14 (b). Although assignment of a state number is performed with the gestalt of this operation like the above-mentioned so that it may have as small a state number as the system state whose task with a high priority is in a Run condition, about the sequence of a state number, it is also possible to carry out so that it may have conversely as large a state number as the system state whose task with a high priority is in a Run condition. The task to which it will be in a Run condition from the range of a state number in any case can be distinguished.

[0071] After generation of the system state minimum table by the condition extract section 22 is completed, the table generation section 23 generates a state-transition table with reference to the system state minimum table and a system definition table (S704). A state-transition table is a table defined that a current system state and the system state after [the generated event to] a state transition should be judged as above-mentioned.

[0072] Drawing 18 and drawing 19 are flow charts which show the detailed contents of processing of the table generation section 23. The table generation section 23 extracts the information on the event relevant to a task and a task with reference to a system definition table first (S901). For example, in the example shown in drawing 11 , the priority of the task to which a change is carried out, and a task is extracted from a task

definition table, and the event relevant to a task is extracted from information, such as the flag ID relevant to the task defined as the synchronous transmission functional definition table, and a mail box ID. Here, it does not necessarily mean extracting to another field as an extract, and even if it refers to a system definition table if needed during generation of a table, it does not interfere.

[0073] Processing of a loop formation A is performed one by one about all the system states included in the system state minimum table (S902). The table generation section 23 reads one system state at a time from the system state minimum table (S903). Processing of a loop formation B is performed about all the events including the case where the task of a Run condition changes to a Wait condition, and the case where the event extracted from the synchronous transmission functional definition table at step S901 occurs that may be generated (S904).

[0074] That is, processing of a loop formation A and a loop formation B is processing which generated the state transition when the system state read in step S903 is assumed to be the present system state and an event occurs in the system state concerned and which simulates for every event and buries every one matrix of a state-transition table like drawing 15 . Hereafter, the contents of the processing in a loop formation B are explained to a detail while an example is shown.

[0075] The table generation section 23 determines whether to simulate about the case where which event occurs, first (S905). Then, it judges whether the event determined in step S905 is the case where the task of a Run condition changes to a Wait condition (S906). This is because it is accompanied by the state transition of a clearly different class by the case where the task of a Run condition changes to a Wait condition, and the case where a task rises according to generating of an event.

[0076] If it is simulation processing of the state transition of an about when the task of a Run condition changes to a Wait condition (S906:Yes), in the selected system state, the task which was in the Run condition will assume first that it is what changes to a Wait condition (S907). For example, in the system state of a state number 1, when assuming the case where the task of a Run condition changed to a Wait condition, Task A should change to the Wait condition first like the example shown in drawing 20 (drawing 20 (b)).

[0077] Next, it judges whether there is any task which is in the Ready condition in the remaining tasks (S908). If there is no task of a Ready condition, it will move to the flow chart of drawing 19 , and the state number of the system state after a state transition will be searched (S909). Here, if the corresponding system state exists in the system state minimum table (S910:Yes), the state number of the corresponding system state is

saved in the location where a state-transition table corresponds (S911). However, when said system state does not exist in the system state minimum table, let the system state after a state transition be an undefined (S912). In this case, the flag (the example of drawing 15 "-") which shows that the state number after a state transition is an undefined is set to the matrix of a state-transition table.

[0078] In step S908, if the task of a Ready condition exists, it will judge whether it moves to the flow chart of drawing 19 , and two or more tasks with the highest priority exist in the task of a Ready condition (S913). If it is the example of drawing 20 , in the condition of drawing 20 (b), the task with the highest priority is Task B in the task of a Ready condition. Therefore, at step S913, it will be judged with No.

[0079] If the number of the tasks of the Ready condition that a priority is the highest is one (S913:No), since the task will be in a Run condition, the system state after a state transition will be determined by making the task of a Ready condition into a Run condition according to generating of an event (S914). In the example of drawing 20 , Task B will be in a Run condition (drawing 20 (c)). When two or more tasks of the Ready condition that a priority is the highest exist, it is necessary to determine which task is made into a Run condition based on the specification of a system etc. For example, although how to change into a Run condition the task which rose at the ** point can be considered, you may determine by other approaches. Anyway, according to a situation, a suitable task will change to a Run condition (S915). Although detailed explanation here is omitted, about a task with the same priority, it is coped with by [which rose to the system state table generate time] giving a definition as another system state for every sequence.

[0080] If the system state after a state transition is determined as mentioned above, the state number corresponding to the determined system state is searched from the system state minimum table (S909) and the corresponding system state exists in the system state minimum table (S910:Yes), the searched state number is saved on a state-transition table (S911). If it is the example of drawing 20 , since the state number of the system state shown in drawing 20 (c) is 4, as shown in drawing 20 (d), a state number 4 will be saved at the position of a state-transition table. In step S910, if a system state does not exist in the system state minimum table, let the system state after a state transition be an undefined (S912).

[0081] On the other hand, when judged with No in step S906 (i.e., when the task of a Run condition will be in a Wait condition), in assuming generating of the event of an except, the task relevant to the event determined at step S905 judges whether it is in a Wait condition in a current system state (S916). Since it is the case where do not mean

that there is no modification in a system state in any way, and a state-transition table is not referred to when it is not in a Wait condition, the system state after a state transition is made into the undefined (S912), and it returns to the head of a loop formation B. Since a task will rise according to generating of an event when it is in a Wait condition, the task of the Wait condition concerned is made into a Ready condition (S917).

[0082] An example of the processing which simulates transition of the system state by generating of an event is shown in drawing 21 . The example shown in this drawing is an example in case Task A rises in the system state of a state number 6 by making a setup of a bit pattern with the flag setting instruction which Task C published to OS field of the event flag which has the event flag ID "FLAG_A." In the example shown in this drawing, Task A will be in a Ready condition by coincidence of the bit pattern of "FLAG_A" (drawing 21 (b)).

[0083] Next, the table generation section 23 measures the priority of the task which changed into the Ready condition at step S917, and the priority of the task which is in a Run condition in a current system state (S918). Since continuation activation of the task of a Run condition is carried out as it is when the priority of the task of a Run condition is higher than the priority of the task of a Ready condition (S918:Yes), if it moves to the flow chart of drawing 19 , the state number showing the system state after a state transition is searched (S909) and a system state exists in the system state minimum table (S910:Yes), a state number is saved at the position of the matrix of a state-transition table (S911).

[0084] Like the example shown in drawing 21 (b), when the priority of the task of a Ready condition is higher than the priority of the task of a Run condition (S918:No), it changes the task of a Run condition into a Ready condition (S919), and progresses to step S913 of the flow chart of drawing 19 . In the example of drawing 21 (b), Task C is changed into a Ready condition in step S919 (drawing 21 (c)), and Task A is made into a Run condition in step S914 after the judgment of step S913 explained previously (drawing 21 (d)).

[0085] If the system state after a state transition is determined as mentioned above, the state number corresponding to the determined system state is searched from the system state minimum table in step S909 and a system state exists in the system state minimum table (S910:Yes), the searched state number is saved on a state-transition table (S911). If it is the example of drawing 21 , since the state number of the system state shown in drawing 21 (d) is 3, as shown in drawing 21 (e), a state number 3 will be saved at the position of a state-transition table.

[0086] A state-transition table as shown in drawing 15 is mechanically generable by what it carries out about all the events that may generate the above processings (S920), and is further performed about all the conditions of being contained in the system state minimum table (S921). In the state-transition table shown in drawing 15 , it means that task switching occurs in the state transition of a half-tone-dot-meshing part.

[0087] In addition, although the gestalt of this operation explains to a detail the system whose condition that each task can take is three kinds, a Wait condition, a Ready condition, and a Run condition, as most general example, other conditions can be taken depending on the function of OS. For example, they are a suspension condition, a suspension standby condition, hibernation, etc. The state transition diagram which included those conditions in drawing 22 is shown. Here, when said each condition is explained briefly, a suspension condition is in the condition that the task is put on the compulsory waiting state. When the task concerned was in a Ready condition and the interruption instruction was published from other tasks, or when waiting is canceled by the setting instruction of a flag etc. in a suspension standby condition, it is in the condition acquired [for] and, specifically, will be in a Ready condition from other tasks by publishing a restart instruction. here -- an interruption instruction -- for example, "suspend (taskID)" -- as -- Task ID is specified and published -- having -- a restart instruction (taskID), for example, "resume", -- as -- although Task ID is specified and it is published, all are accompanied by change of a system state.

[0088] A suspension standby condition is in the condition acquired [for], when an interruption instruction is published in a Wait condition, and it changes in the suspension condition by discharge of waiting by generating of an event like the above-mentioned. Moreover, when a restart instruction is published from other tasks, it returns to a Wait condition. Hibernation is in the condition of having also opened the minimum resource for task activation, such as a stack, wide, and having stopped activation of a task. To hibernation, when a forced-termination instruction is published from conditions, such as a case where a termination instruction is published in [a task's] the own one of a Run condition, and a Ready condition, a Wait condition, it changes. Moreover, when a starting instruction is published from other tasks from hibernation, from it, it shifts to a Ready condition.

[0089] A termination instruction makes the task of the Run condition concerned itself change to hibernation here, when the task of a Run condition itself publishes an instruction, such as "exittask". an interruption instruction -- for example, "terminate (taskID)" -- as -- Task ID is specified and published -- having -- a starting instruction -- for example, "starttask (taskID)" -- as -- Task ID is specified and it is published. All are

accompanied by change of a system state.

[0090] However, so that clearly from the state transition diagram shown in drawing 22 , and the above explanation In the point that a state transition happens also about which condition according to generating of the event based on the system call instruction at the time of driving a multitasking control unit Said Wait, It is the same on three kinds of conditions and the essential target of Ready and Run, and applying easily is possible by extending [generation / of a state-transition table] a little the approach explained to the detail this time also about the task switching using a state-transition table.

[0091] That is, if it is at the time of task switching, it considers that issue of said interruption instruction, a restart instruction, etc. is generating of an event, and the system state after a state transition is judged by referring to a state-transition table, and the task which should be further performed after a state transition from the system state after a state transition is judged. In generation of a state-transition table, what is necessary is just made to perform simulation processing of the state transition at the time of event generating which was explained in the gestalt of this operation also with said interruption instruction and a restart instruction based on a state transition diagram.

[0092] Moreover, although considered as the form where the state number which expresses the system state after each event occurs for every state-number number as a state-transition table is held, with the gestalt of this operation, it does not matter even if it holds directly the table which a format of a table is not restricted to the example of the gestalt of this operation, but may carry out it to other formats as long as it can determine the task which should be performed after a state transition, for example, expresses a system state.

[0093] Moreover, although it was made the form where gave priority to improvement in the speed of processing, and the information about the task under current activation was held in the task state data-hold section 15, with the gestalt of this operation, it is also possible like the above-mentioned to distinguish the task under current activation from the state number currently held at the scheduling Management Department 17. Moreover, although the state number in the time was held to the scheduling Management Department 17 with the gestalt of this operation, if a system state can be recognized, it is not necessary to necessarily hold in the form of a state number. Moreover, you may make it the information for identifying a system state also detect a system state at the time of generating of an event, without always holding.

[0094] Moreover, also in the condition extract processing by the condition extract section 22, it is also possible to generate the system state minimum table based on the

regulation about system states other than the regulation about the system state used with the gestalt of this operation. Moreover, although it is made to perform those processings of all each time with the gestalt of this operation about the processing performed in the pretreatment section 11, the processing which does not necessarily need to be performed each time may be included in the processing which should be performed in the pretreatment section 11. About such processing, only when change of a system state is detected by each event Management Department, for example, it is also possible for it to be made to carry out after processing of each event Management Department. In that case, it is not necessary to carry out also in the after-treatment section 19 about the after treatment corresponding to the processing which was not performed as processing of the pretreatment section 11.

[0095]

[Effect of the Invention] The multitasking control unit concerning this invention so that clearly from the above explanation In the multitasking control unit which changes the task which prepares for the system which performs two or more tasks, and said system performs according to generating of an event For every system state showing in which condition in various kinds of conditions including a running state, ready condition, and a standby condition of two or more of said tasks it is, respectively The state-transition table which holds the information which determines the task which should be performed next when each event occurs, A judgment means to judge the task which should refer to said state-transition table and should be performed next from the system state at the time when generating of an event is detected by event detection means to detect generating of an event, and said event detection means, It has a task switching means to change a task according to the judgment of said judgment means. A judgment means When generating of an event is detected by said event detection means, the state-transition table holding the information which determines the task which should be performed next when each event occurs is referred to for every system state. Since the task which should be performed next from the system state at the time is judged At the time of generating of an event, queue actuation of the search of a state-transition queue, queue connection, etc., Comparison processing of the priority of the task of a Ready condition with the task of a Run condition, comparison processing of the priority of the tasks of a Ready condition, and the state-transition queue at the time of task switching retie, and it is not necessary to perform actuation etc., and is effective in becoming possible to perform scheduling processing at a high speed.

[0096] Moreover, said judgment means can also have an attaching part holding the system state at the time. In that case, at the time of generating of each event, since it is

not necessary to detect the system state in the time, there is effectiveness that scheduling processing can be performed more at a high speed. Moreover, said state-transition table can hold the system state of a transition place when each event occurs for every system state about the system state which may exist in said system. While being able to save the field which holds a state-transition table by holding only about the system state in which said state-transition table may exist in said system, it is effective in the ability to recognize quickly, without investigating each task status for the system state after a state transition by holding the system state of a transition place when each event occurs for every system state.

[0097] Moreover, in said state-transition table, to each system state, a state number is assigned in the system state whose task with a high priority is a running state to ascending order or descending order, and said judgment means can judge the task which should be performed next from the state number of the system state of the transition place currently held at the state-transition table. After extract processing of a system state, if the state number is assigned like the above while being able to save the field which holds a state-transition table by reassigning a state number to a system state, the effectiveness that said judgment means can judge easily which task will be in a Run condition is.

[0098] Moreover, it can also have further a state-transition table generation means to generate said state-transition table, from a system definition table including the definition with a task, the priority for every task, and the event that rises each task to ready condition. In this case, when a state-transition table generation means generates a state-transition table mechanically from a system definition table, it is effective in the ability to prevent the generation mistake of a state-transition table.

[0099] Moreover, a condition generation means by which said state-transition table generation means generates all the combination of each task status from said system definition table, By deleting the combination showing the system state which cannot exist in said system from the combination which said condition generation means generated according to the regulation about a predetermined system state It is possible to have a condition extract means to extract the combination showing the system state which may exist in said system, and a table generation means to generate said state-transition table from the combination extracted by said condition extract means.

[0100] A condition generation means generates all the combination of each task status with reference to a system definition table. A condition extract means By deleting the combination showing the system state which cannot exist in a system from all the combination of each of said task status The combination showing the system state

which may exist in a system is extracted. A table generation means By generating a state-transition table from the combination extracted by said condition extract means Generating a state-transition table about the system state which cannot exist in a system is lost, and it becomes possible to save the field holding a state-transition table, and also is effective in the ability to prevent the generation mistake of a state-transition table with an easy algorithm.

[0101] Moreover, said condition extract means can extract the combination which expresses said system state according to the regulation about the system state based on the function of said system, and the regulation about the system state based on the reason of an application proper. By performing such processing, it has the effectiveness that it can respond to modification of the function of an O ** rating system, and modification of application flexibly.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] It is the block diagram showing the configuration of the multitasking control device in the gestalt of 1 operation of this invention.

[Drawing 2] It is the flow chart which shows the contents of processing of the multitasking control device in the gestalt of 1 operation of this invention.

[Drawing 3] It is the flow chart which shows the detailed contents of processing of flag setting instruction processing which the event flag Management Department performs.

[Drawing 4] It is the flow chart which shows the detailed contents of processing of waiting instruction processing for a flag which the event flag Management Department performs.

[Drawing 5] It is the flow chart which shows the detailed contents of processing of the resource deallocation demand processing which the semaphore Management Department performs.

[Drawing 6] It is the flow chart which shows the detailed contents of processing of the resource acquisition demand processing which the semaphore Management Department performs.

[Drawing 7] It is the flow chart which shows the detailed contents of processing of the e-mail transmitting processing which the mail administration section performs.

[Drawing 8] It is the flow chart which shows the detailed contents of processing of the e-mail reception which the mail administration section performs.

[Drawing 9] It is the flow chart which shows the detailed contents of processing of the scheduling Management Department.

[Drawing 10] It is the block diagram showing the configuration of the state-transition table generation section.

[Drawing 11] (a) It is drawing showing an example of the contents of the task definition table contained in a system definition table.

(b) It is drawing showing an example of the contents of the synchronous transmission functional definition table contained in a system definition table.

[Drawing 12] It is drawing showing the DS of a system state table.

[Drawing 13] (a) It is drawing showing an example of OS regulation.

(b) It is drawing showing an example of an application proper regulation.

[Drawing 14] (a) It is drawing showing an example of the system state table after the condition extract by the condition extract section.

(b) It is drawing showing an example of the system state minimum table.

[Drawing 15] It is drawing showing an example of a state-transition table.

[Drawing 16] It is the flow chart which shows the contents of processing of the state-transition table generation section.

[Drawing 17] It is the flow chart which shows the contents of processing of the condition extract processing based on OS regulation.

[Drawing 18] It is the flow chart which shows the contents of processing of the table generation section.

[Drawing 19] It is the flow chart which shows the contents of processing of the table generation section.

[Drawing 20] It is drawing for explaining the contents of processing of the table generation section.

[Drawing 21] It is drawing for explaining the contents of processing of the table generation section.

[Drawing 22] It is a state transition diagram including a suspension condition, a suspension standby condition, and hibernation.

[Drawing 23] It is the block diagram showing the configuration of the conventional multitasking control device.

[Drawing 24] It is drawing for explaining actuation of the conventional multitasking control unit.

[Description of Notations]

11 Pretreatment Section

12 Event Flag Management Department

13 Semaphore Management Department

14 Mail Administration Section

15 Task State Data-hold Section

16 State-Transition Table Attaching Part

17 Scheduling Management Department

18 Task Switching Management Department

19 After-Treatment Section

20 State-Transition Table Generation Section

21 Condition Generation Section

22 Condition Extract Section

221 Condition Extract Section Based on OS Regulation

222 Condition Extract Section Based on Application Proper Regulation

23 Table Generation Section

30 System Definition Table Attaching Part

---

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平9－319595

(43)公開日　平成９年(1997)12月12日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　9/46	３４０		Ｇ０６Ｆ　9/46	３４０Ｂ	

審査請求　未請求　請求項の数７　ＯＬ　（全 21 頁）

(21)出願番号　特願平8－131484

(22)出願日　平成８年(1996)５月27日

(71)出願人　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地

(72)発明者　岩村　喜之
大阪府門真市大字門真1006番地　松下電器産業株式会社内

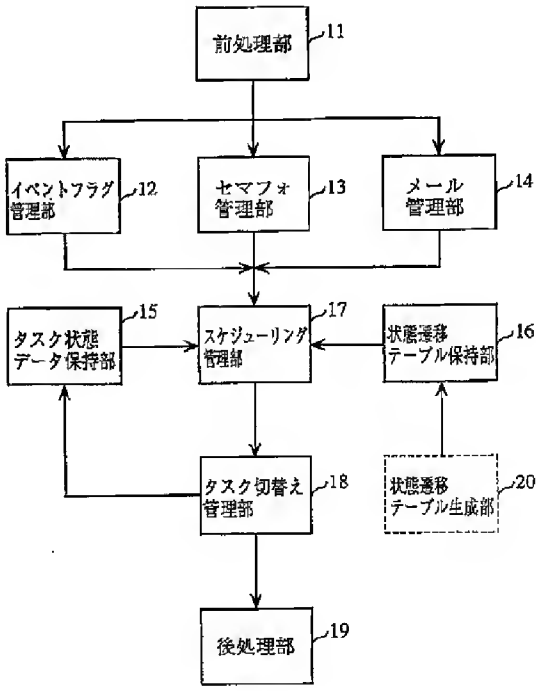
(72)発明者　角　史生
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　西畑　素秀
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(74)代理人　弁理士　中島　司朗

(54)【発明の名称】　マルチタスク制御装置

(57)【要約】
【課題】　高速なスケジューリング処理を可能とするマルチタスク制御装置を提供する。
【解決手段】　前処理部１１、イベントフラグ管理部１２、セマフォ管理部１３、メール管理部１４、タスク状態データ保持部１５、状態遷移テーブル保持部１６、スケジューリング管理部１７、タスク切替え管理部１８、後処理部１９、状態遷移テーブル生成部２０を備え、スケジューリング管理部１７が、イベントフラグ管理部１２、セマフォ管理部１３、メール管理部１４からの事象の発生の通知を受けて、タスク状態データ保持部１５の内容と、状態遷移テーブル保持部１６に保持されている状態遷移テーブルとを参照し、事象の発生後に実行すべきタスクを判定して、必要に応じてタスク切替え管理部１８にタスクの切替えを指示する。



1

【特許請求の範囲】

【請求項1】　複数のタスクを実行するシステムに備えられ、事象の発生に応じて、前記システムが実行するタスクの切替えを行うマルチタスク制御装置において、

前記複数のタスクのそれぞれが、実行状態、実行可能状態、待機状態を含む各種の状態の中のどの状態にあるかを表すシステム状態ごとに、各事象が発生した場合に次に実行すべきタスクを決定する情報を保持する状態遷移テーブルと、

事象の発生を検知する事象検知手段と、

前記事象検知手段により事象の発生が検知された場合に、前記状態遷移テーブルを参照し、その時点におけるシステム状態から次に実行すべきタスクを判定する判定手段と、

前記判定手段の判定に応じてタスクの切替えを行うタスク切替え手段とを有することを特徴とするマルチタスク制御装置。

【請求項2】　前記判定手段は、

その時点におけるシステム状態を保持する保持部を有することを特徴とする請求項1記載のマルチタスク制御装置。

【請求項3】　前記状態遷移テーブルは、

前記システムにおいて存在し得るシステム状態について、システム状態ごとに、各事象が発生した場合の遷移先のシステム状態を保持することを特徴とする請求項1又は2記載のマルチタスク制御装置。

【請求項4】　前記状態遷移テーブルにおいて、

各々のシステム状態には、優先度の高いタスクが実行状態であるシステム状態から、昇順又は降順にて状態番号が割り振られ、

前記判定手段は、

状態遷移テーブルに保持されている遷移先のシステム状態の状態番号から次に実行すべきタスクを判定することを特徴とする請求項3記載のマルチタスク制御装置。

【請求項5】　前記マルチタスク制御装置はさらに、

タスクと、タスクごとの優先度と、各々のタスクを実行可能状態に起床させる事象との定義を含むシステム定義テーブルから、前記状態遷移テーブルを生成する状態遷移テーブル生成手段を備えることを特徴とする請求項1乃至4の何れかに記載のマルチタスク制御装置。

【請求項6】　前記状態遷移テーブル生成手段は、

前記システム定義テーブルから、各タスクの状態の全ての組み合わせを生成する状態生成手段と、

所定のシステム状態に関する規則に従って、前記状態生成手段が生成した組み合わせから、前記システムにおいて存在し得ないシステム状態を表す組み合わせを削除することにより、前記システムにおいて存在し得るシステム状態を表す組み合わせを抽出する状態抽出手段と、

前記状態抽出手段により抽出された組み合わせから、前記状態遷移テーブルを生成するテーブル生成手段とを有

2

することを特徴とする請求項5記載のマルチタスク制御装置。

【請求項7】　前記状態抽出手段は、

前記システムの機能に基づくシステム状態に関する規則と、アプリケーション固有の理由に基づくシステム状態に関する規則とに従って前記システム状態を表す組み合わせを抽出することを特徴とする請求項6記載のマルチタスク制御装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、オペレーティングシステム（以下「OS」と略称する）によるマルチタスク制御を行うマルチタスク制御装置に関する。

【0002】

【従来の技術】近年、マイコン制御を行う家電製品等の分野において、複数のタスクを有するマルチタスクプログラムの実行に際して、同期通信機能を用いてタスクの切替えを行うマルチタスク制御装置が広く利用されている。ここで、同期通信機能とは、複数のタスクの処理を同期して進行させるためにタスク間で通信を行う機能のことである。以下、同期通信機能を用いた従来のマルチタスク制御装置について説明する。

【0003】図23は、従来のマルチタスク制御装置の構成を示すブロック図である。同図に示されるマルチタスク制御装置は、前処理部91、イベントフラグ管理部92、セマフォ管理部93、メール管理部94、状態遷移キュー95、スケジューリング管理部96、タスク切替え管理部97、後処理部98を有する。前処理部91は、マルチタスク制御装置が、タスク若しくは割り込みルーチンからシステム呼び出し命令によって駆動された直後に、呼び出し元を表す識別子の保存処理等、前処理として毎回行う必要のある処理を行う。

【0004】ここで、呼び出し元を表す識別子の保存を行う必要性は、呼び出し元によって復帰時の処理内容に相違があることに起因する。即ち、マルチタスク制御装置がタスクから駆動された場合には、復帰時にタスクの切替えを行っても何ら問題は起きない。しかし、割り込みルーチンから駆動された場合には、割り込み処理を全て終了してタスクに制御が戻るときにはタスクの切替えを行ってもよいが、それ以外の場合、例えば多重割り込みが行われた状態で割り込みルーチンから駆動された場合のように、戻り先も割り込みルーチンであるような場合にはタスクの切替えを行うことはできない。これは、戻り先が割り込みルーチンである場合にタスクの切替えを行ってしまうと、本来の戻り先である呼び出し元の割り込みルーチンに戻れなくなってしまうというOSの機能に基づくものである。

【0005】従って、マルチタスク制御装置からの復帰時には、呼び出し元を識別して、呼び出し元に応じた処理を行う必要があるため、呼び出し元を表す識別子の保

存処理が必要となるのである。イベントフラグ管理部９２は、マルチタスク制御装置がタスク若しくは割り込みルーチンからイベントフラグの設定命令又は待ち命令により駆動された場合にイベントフラグの設定処理等を行う。

【０００６】ここで、イベントフラグとは、ＯＳが管理するデータ領域（以下「ＯＳ領域」と呼ぶ）に、複数のフラグＩＤごとに保持されているビットパタンを意味する。イベントフラグの設定命令とは、タスク若しくは割り込みルーチンからフラグＩＤとビットパタンとを指定してイベントフラグの設定を行う命令である。具体的には、例えばタスク若しくは割り込みルーチンにおいて "setflag(setpattern,flagID)" というような命令が発行されると、イベントフラグ管理部９２によりflagIDで指定されたフラグＩＤを有するＯＳ領域にsetpatternで指定されたビットパタンが設定される。設定命令は、タスク及び割り込みルーチンの両方から発行される場合がある。

【０００７】一方、イベントフラグの待ち命令とは、タスクからフラグＩＤとビットパタンを指定して、当該フラグＩＤを有するＯＳ領域に設定されているビットパタンと、タスクにより指定されたビットパタンとが一致するか否かを問い合わす命令である。具体的には、例えばタスクにおいて "waitflag(waitpattern,flagID)" というような命令が発行されると、イベントフラグ管理部９２はflagIDで指定されたＩＤを有するＯＳ領域に設定されているビットパタンと、waitpatternで指定されたビットパタンとを比較する。比較の結果、両者が一致していれば命令を発行したタスクをそのまま処理を実行することが可能であるが、一致していなければ待機状態（以下「Ｗａｉｔ状態」と呼ぶ）となる。Ｗａｉｔ状態とは、事象の発生によりタスクが実行可能状態（以下「Ｒｅａｄｙ状態」と呼ぶ）となるまで（以下、Ｗａｉｔ状態のタスクが事象の発生によりＲｅａｄｙ状態となることを「起床する」と称する）処理の実行を中断している状態である。この場合の例であれば、他のタスク若しくは割り込みルーチンから発行されたイベントフラグの設定命令により、両者のビットパタンが一致するするまでＷａｉｔ状態が継続する。なお、イベントフラグの待ち命令はタスクのみから発行され、割り込みルーチンから発行されることはない。

【０００８】Ｗａｉｔ状態において、事象の発生によりタスクが起床すると、当該タスクはまずＲｅａｄｙ状態となり、当該タスクの優先度が最も高い場合や、他にＲｅａｄｙ状態のタスクが存在しない場合等、当該タスクを実行することが可能な状況が整えば実行状態（以下「Ｒｕｎ状態」と呼ぶ）となる。イベントフラグの待ち命令において、waitpatternで指定されたビットパタンは、個々のタスクに割り当てられたＴＣＢ(Task Control Block)に、フラグＩＤの指定と共に保持される。

【０００９】イベントフラグ管理部９２は、マルチタスク制御装置がイベントフラグの設定命令により駆動された場合には、イベントフラグの設定処理後、当該フラグＩＤを有するフラグに対してビットパタンを指定してＷａｉｔ状態となっているタスクが存在するか否かを検索する。検索の結果Ｗａｉｔ状態のタスクが存在し、かつ、Ｗａｉｔ状態のタスクにより指定されているビットパタンと、設定処理後のＯＳ領域のビットパタンとが一致した場合にはＷａｉｔ状態のタスクが起床する。この場合、イベントフラグ管理部９２は、事象の発生によるタスクの起床をスケジューリング管理部９６に通知する。

【００１０】また、イベントフラグ管理部９２は、マルチタスク制御装置がイベントフラグの待ち命令により駆動された場合には、前述のとおりタスクにより指定されたビットパタンとＯＳ領域のビットパタンとを比較し、一致していればそのままタスクの処理の実行を継続する。一致しない場合には、タスクがＷａｉｔ状態となるため、イベントフラグ管理部９２はタスクのＷａｉｔ状態への変化をスケジューリング管理部９６に通知する。

【００１１】セマフォ管理部９３は、資源に割り振られたＩＤごとに、その時点での資源の割当可能量を保持するカウンタを有しており、マルチタスク制御装置が、タスク若しくは割り込みルーチンから資源獲得要求及び資源開放要求により駆動された場合にカウンタの更新処理等を行う。ここで、資源獲得要求とはタスクから資源のＩＤと資源の獲得要求量を指定して発行される命令である。具体的には、例えば "waitsem(semID,reqNo)" というような命令が発行されると、セマフォ管理部９３はreqNoで指定された要求量と、semIDで指定されたＩＤを有する資源についてのカウンタに保持されている割当可能量とを比較する。割当可能量が獲得要求量以上であれば、セマフォ管理部９３がカウンタの値から獲得要求量を減算し、タスクに対して、要求された資源が割り当てられて、当該タスクの処理が継続実行されるが、割当可能量が獲得要求量を下回っている場合には、当該タスクは、他のタスクからの資源開放要求により当該資源の割当可能量が増加して獲得要求量以上となるまでＷａｉｔ状態となる。

【００１２】一方、資源開放要求とは、タスクから資源のＩＤと資源の開放要求量を指定して発行される命令である。具体的には、例えば "sigsem(semID,relNo)" というような命令が発行されると、semIDで指定されたＩＤを有する資源についてのカウンタの値を加算する。なお、資源開放要求は、タスク及び割り込みルーチンの両方から発行される可能性があるが、資源獲得要求はタスクのみから発行され、割り込みルーチンから発行されることはない。

【００１３】セマフォ管理部９３は、マルチタスク制御装置が資源開放要求により駆動された場合には、カウン

タの値の加算の後、当該資源の割当可能量の増加を待ってＷａｉｔ状態となっているタスクが存在するか否かを検索する。検索の結果Ｗａｉｔ状態のタスクが存在し、かつ、Ｗａｉｔ状態のタスクにより要求されている資源量が、加算後のカウンタの値以下である場合には、Ｗａｉｔ状態のタスクが起床する。この場合セマフォ管理部９３は、事象の発生によるタスクの起床をスケジューリング管理部９６に通知する。

【００１４】セマフォ管理部９３は、マルチタスク制御装置が資源獲得要求により駆動された場合には、前述のとおりカウンタに保持されている資源の割当可能量が獲得要求量以上であれば、カウンタの値を減算して、そのままタスクの処理の実行を継続する。割当可能量が獲得要求量を下回る場合は当該タスクがＷａｉｔ状態となるため、セマフォ管理部９３は、タスクのＷａｉｔ状態への変化をスケジューリング管理部９６に通知する。

【００１５】メール管理部９４は、マルチタスク制御装置がタスク若しくは割り込みルーチンからメールの受信命令又は送信命令により駆動された場合にメールの管理処理等を行う。メールは複数のメールボックスＩＤごとに、メッセージを伴って、送信された順に保持される。メールの受信命令とは、タスクからメールボックスＩＤを指定して当該メールボックスにメールが保持されているか否かを問い合わす命令である。具体的には、例えば"waitmsg(boxID,ptr)"というような命令が発行されると、boxIDで指定されたメールボックスにメールが保持されているか否かを検索する。メールが保持されていれば、受信命令を発行したタスクは、そのメールを受信して処理の実行を継続することができる。ptrには、メッセージが存在する位置のアドレスが返されるので、タスクはメッセージ領域に保持されたデータを使用して処理を継続する。

【００１６】メールが保持されていなければ、タスクは、他のタスクから当該メールボックスにメールが送信されるまでＷａｉｔ状態となる。また、メールの送信命令とは、タスク若しくは割り込みルーチンから、メールボックスＩＤとメッセージ保持領域のアドレスとを指定してメールを送信する命令である。具体的には"sendmsg(boxID,ptr)"というような命令が発行されると、boxIDで指定されたメールボックスにptrで指定されたアドレスに保持されるメッセージを伴うメールが送信される。

【００１７】なお、メールの送信命令は、タスク及び割り込みルーチンの両方から発行される可能性があるが、メールの受信命令はタスクのみから発行され、割り込みルーチンから発行されることはない。メール管理部９４は、マルチタスク制御装置がメールの送信命令により駆動された場合には、メールの送信処理を行った後、当該メールボックスへのメールの送信を待ってＷａｉｔ状態となっているタスクが存在するか否かを検索する。

【００１８】検索の結果Ｗａｉｔ状態のタスクが存在していれば、Ｗａｉｔ状態のタスクが起床することになるので、メール管理部９４は、事象の発生によるタスクの起床をスケジューリング管理部９６に通知する。メール管理部９４は、マルチタスク制御装置がメールの受信命令により駆動された場合には、前述のとおり指定されたメールボックスにメールが保持されていれば、タスクにメールを渡して、そのままタスクの処理の実行を継続する。メールが保持されていない場合は、受信命令を発行したタスクがＷａｉｔ状態となるため、メール管理部９４は、タスクのＷａｉｔ状態への変化をスケジューリング管理部９６に通知する。

【００１９】状態遷移キュー９５は、ＲｕｎキューとＲｅａｄｙキューとから構成される。ＲｕｎキューとはＲｕｎ状態のタスクのＴＣＢが接続されている領域であり、Ｒｅａｄｙキューとは、Ｒｅａｄｙ状態のタスクが、例えば優先順位の順に待ち行列を形成している領域である。スケジューリング管理部９６は、イベントフラグ管理部９２、セマフォ管理部９３、メール管理部９４からの通知を受け、Ｒｕｎ状態からＷａｉｔ状態となったタスクをＲｕｎキューから切り離したり、事象の発生により起床したタスクについて、Ｒｅａｄｙ状態の各々のタスクの優先度を比較しながら、Ｒｅａｄｙキューの待ち行列の並び順を管理するとともに、優先度の高いタスクから順にＲｕｎキューに接続されるようにスケジューリングを行う。

【００２０】タスク切替え管理部９７は、スケジューリング管理部９６のスケジューリングに従って、実際にレジスタの退避／復帰等のタスクの切替え処理を行う。後処理部９８は、前処理部９１において保存された呼び出し元の識別子等の更新、削除等、マルチタスク制御の最後に毎回行う必要のある処理を行う。以上のように構成されたマルチタスク制御装置について、以下、その動作を具体例を用いて説明する。図２４は、本具体例における状態遷移キュー９５の状態を示す図である。

【００２１】例えば、三つのタスクＡ、Ｂ、Ｃが存在し、タスクＡが最高優先度、タスクＢが次の優先度、タスクＣが最低優先度であり、タスクＡがＷａｉｔ状態、タスクＢがＲｕｎ状態、タスクＣがＲｅａｄｙ状態である場合を仮定する。タスクＡはメールボックスＸにメールが送信されるのを待っているＷａｉｔ状態であるとして、Ｒｕｎ状態であるタスクＢからメールボックスＸに対するメールの送信命令が発行された場合の動作について説明する。

【００２２】まず前処理部９１は、タスクＢからメールの送信命令により駆動されたことを表す識別子の保存等のＯＳ共通処理を行う。今回の例ではメールの送信命令により呼び出されているのでメール管理部９４による処理が行われる。メール管理部９４は、メールボックスＸにメールを送信した後、メールボックスＸへのメールの送信を待ってＷａｉｔ状態となっているタスクが存在す

るか否かを検索する。検索の結果、タスクAが起床してReady状態となることが判明するので、メール管理部94は、メールの送信によるタスクの起床をスケジューリング管理部96に通知する。

【0023】スケジューリング管理部96は、メール管理部94の通知を受け、まず状態遷移キュー95の状態を図24（a）から図24（b）のように変更する。即ち、Wait状態であったタスクAがReady状態となるのでタスクAのTCBをReadyキューに繋ぐ。この際、タスクAの優先度はタスクCの優先度より高いので、Readyキューのサーチの結果、タスクA、タスクCの順にReadyキューに繋がれることになる。

【0024】次にスケジューリング管理部96はRun状態であるタスクBと、Ready状態であるタスクの中で最も優先度の高いタスクであるタスクAとの優先度を比較し、状態遷移キュー95の状態を図24（b）から図24（c）のように変更する。即ち、タスクAの優先度の方がタスクBの優先度より高いので、タスクAのTCBをRunキューに繋ぎ、タスクBのTCBをReadyキューに繋ぐ。この場合、タスクBの優先度はタスクCの優先度よりも高いのでReadyキューのサーチの結果タスクB、タスクCの順にReadyキューに繋がれる。

【0025】スケジューリング管理部96は、Runキューに繋がれたタスクの変化を検知し、タスクの切替えが必要であることをタスク切替え管理部97に通知する。タスク切替え管理部97は、レジスタの退避／復帰等、実際のタスク切替えに必要な処理を行い、後処理部98に制御を渡す。後処理部98が、タスクBから駆動されたことを示す識別子の削除等のOS共通処理を行った後、マルチタスク制御装置の処理が終了する。処理終了後はタスクAの処理が実行される。

【0026】

【発明が解決しようとする課題】しかしながら、上記のような従来のマルチタスク制御装置においては、イベントフラグの設定、メールの送信等の事象の発生によりタスクのWait状態が解除され、Ready状態となった場合に、キューサーチ、キュー接続などのキュー操作が必要であること、及び、Ready状態となったタスクとRun状態のタスクとの優先度の比較処理、Ready状態のタスク同士の優先度の比較処理、タスク切替え時のRunキュー、Readyキューの操作等が必要であり、スケジューリング処理が遅いという問題点を有していた。その結果、例えば高速な制御処理を行う必要がある場合には、高機能を有する汎用的なマルチタスクOSを使用することができない等の問題が生じていた。

【0027】本発明は上記の問題点に鑑み、高速なスケジューリング処理を可能とするマルチタスク制御装置を提供することを目的とする。

【0028】

【課題を解決するための手段】上記の問題点を解決する目的で、本発明に係るマルチタスク制御装置は、複数のタスクを実行するシステムに備えられ、事象の発生に応じて、前記システムが実行するタスクの切替えを行うマルチタスク制御装置において、前記複数のタスクのそれぞれが、実行状態、実行可能状態、待機状態を含む各種の状態の中のどの状態にあるかを表すシステム状態ごとに、各事象が発生した場合に次に実行すべきタスクを決定する情報を保持する状態遷移テーブルと、事象の発生を検知する事象検知手段と、前記事象検知手段により事象の発生が検知された場合に、前記状態遷移テーブルを参照し、その時点におけるシステム状態から次に実行すべきタスクを判定する判定手段と、前記判定手段の判定に応じてタスクの切替えを行うタスク切替え手段とを有する。

【0029】また、前記判定手段は、その時点におけるシステム状態を保持する保持部を有することもできる。また、前記状態遷移テーブルは、前記システムにおいて存在し得るシステム状態について、システム状態ごとに、各事象が発生した場合の遷移先のシステム状態を保持することができる。

【0030】さらに、前記状態遷移テーブルにおいて、各々のシステム状態には、優先度の高いタスクが実行状態であるシステム状態から、昇順又は降順にて状態番号が割り振られ、前記判定手段は、状態遷移テーブルに保持されている遷移先のシステム状態の状態番号から次に実行すべきタスクを判定することが可能である。前記マルチタスク制御装置はさらに、タスクと、タスクごとの優先度と、各々のタスクを実行可能状態に起床させる事象との定義を含むシステム定義テーブルから、前記状態遷移テーブルを生成する状態遷移テーブル生成手段を備えることもできる。

【0031】また、前記状態遷移テーブル生成手段は、前記システム定義テーブルから、各タスクの状態の全ての組み合わせを生成する状態生成手段と、所定のシステム状態に関する規則に従って、前記状態生成手段が生成した組み合わせから、前記システムにおいて存在し得ないシステム状態を表す組み合わせを削除することにより、前記システムにおいて存在し得るシステム状態を表す組み合わせを抽出する状態抽出手段と、前記状態抽出手段により抽出された組み合わせから、前記状態遷移テーブルを生成するテーブル生成手段とを有することも可能である。

【0032】さらに、前記状態抽出手段は、前記システムの機能に基づくシステム状態に関する規則と、アプリケーション固有の理由に基づくシステム状態に関する規則とに従って前記システム状態を表す組み合わせを抽出することが可能である。

【0033】

【発明の実施の形態】以下、本発明の一実施の形態につ

いて図面を参照しながら説明する。図1は、本実施の形態に係るマルチタスク制御装置の構成を示すブロック図である。同図に示されるマルチタスク制御装置は、前処理部11、イベントフラグ管理部12、セマフォ管理部13、メール管理部14、タスク状態データ保持部15、状態遷移テーブル保持部16、スケジューリング管理部17、タスク切替え管理部18、後処理部19を有する。

【0034】なお、本実施の形態のマルチタスク制御装置は、OSをCPUが実行することにより実現されている点、及び、CPUにおいて、イベントフラグの設定命令及び待ち命令、資源の獲得要求及び開放要求、メールの送信命令及び受信命令等の命令が実行される毎に駆動されるものである点については、従来の技術で説明したマルチタスク制御装置と同一である。

【0035】前処理部11は、本実施の形態のマルチタスク制御装置が、タスク若しくは割り込みルーチンから駆動された直後に、呼び出し元を表す識別子の保存処理等、前処理として毎回行う必要のある処理を行う。イベントフラグ管理部12は、マルチタスク制御装置がタスク若しくは割り込みルーチンからイベントフラグの設定命令又は待ち命令により駆動された場合に、イベントフラグに関する事象の発生及び当該事象の発生に伴うシステム状態の変化を管理する。システム状態とは、各々のタスクがRun状態、Ready状態、Wait状態の何れの状態であるかを示す情報である。

【0036】具体的には、マルチタスク制御装置が、イベントフラグの設定命令で駆動された場合には、イベントフラグの設定処理を行う他、当該フラグの設定によりシステム状態に変化があった場合に、設定命令で指定されたフラグIDを有するフラグの設定という事象の発生をスケジューリング管理部17に通知する。また、マルチタスク制御装置が、イベントフラグの待ち命令で駆動された場合には、駆動したタスクがWait状態に変化した場合に、マルチタスク制御装置を駆動したタスクのWait状態への変化という事象の発生をスケジューリング管理部17に通知する。

【0037】セマフォ管理部13は、マルチタスク制御装置が、タスク若しくは割り込みルーチンから資源の獲得要求又は資源開放要求により駆動された場合に、資源の制御に関する事象の発生及び当該事象の発生に伴うシステム状態の変化を管理する。具体的には、マルチタスク制御装置が、資源の開放要求で駆動された場合には、カウンタの加算処理を行う他、当該資源に開放によりシステム状態に変化があった場合に、資源の開放という事象の発生をスケジューリング管理部17に通知する。また、マルチタスク制御装置が、資源の獲得要求で駆動された場合には、駆動したタスクがWait状態に変化した場合に、マルチタスク制御装置を駆動したタスクのWait状態への変化という事象の発生をスケジューリング管理部17に通知する。

【0038】メール管理部14は、マルチタスク制御装置が、タスク若しくは割り込みルーチンからメールの受信命令又は送信命令により駆動された場合に、メールに関する事象の発生及び当該事象の発生に伴うシステム状態の変化を管理する。具体的には、マルチタスク制御装置がメールの送信命令で駆動された場合には、当該メールの送信でシステム状態に変化があった場合に、送信命令で指定されたメールボックスIDを有するメールボックスへのメールの送信という事象の発生をスケジューリング管理部17に通知する。また、マルチタスク制御装置が、メールの受信命令で駆動された場合には、駆動したタスクがWait状態に変化した場合に、マルチタスク制御装置を駆動したタスクのWait状態への変化という事象の発生をスケジューリング管理部17に通知する。

【0039】タスク状態データ保持部15には、実行中のタスクを示す識別子が保持されている。状態遷移テーブル保持部16は、現在のシステム状態と、発生した事象とから、状態遷移後のシステム状態を判定するための状態遷移テーブルを保持する。状態遷移テーブルは、状態遷移テーブル生成部20にて予め生成される。状態遷移テーブルのデータ構造及び生成方法については後述する。

【0040】スケジューリング管理部17は、イベントフラグ管理部12、セマフォ管理部13、メール管理部14からの通知を受け、タスク状態データ保持部15及び状態遷移テーブル保持部16の内容を参照して、タスク切替えが必要か否かを判定し、タスク切替えが必要な場合にタスク切替え管理部18に指示する。タスク切替え管理部18は、スケジューリング管理部17の指示に従って、レジスタの退避／復帰等のタスクの切替え処理を行うとともに、タスク状態データ保持部15の内容を更新する。

【0041】後処理部19は、前処理部11において保存された呼び出し元の識別子等の更新、削除等、マルチタスク制御の最後に毎回行う必要のある処理を行う。図2は本実施の形態のマルチタスク制御装置の処理内容を示すフローチャートである。本実施の形態のマルチタスク制御装置は、タスク若しくは割り込みルーチンからのシステム呼び出し命令を受けて動作を開始する（S101）。

【0042】まず、前処理部11がシステム呼び出しを行ったタスク若しくは割り込みルーチンの識別子を保存する等のOS共通処理を行う（S102）。次に、システム呼び出し命令の内容に基づき、イベントフラグ管理部12、セマフォ管理部13、メール管理部14の何れかによる処理が行われる。例えば、システム呼び出し時の命令がイベントフラグの設定命令であれば、イベントフラグ管理部12により、フラグ設定命令処理が行われ

る（Ｓ１０４）。また、イベントフラグの待ち命令であれば、イベントフラグ管理部１２により、フラグ待ち命令処理が行われる（Ｓ１０５）。

【００４３】また、システム呼び出し時の命令が資源の開放要求であれば、セマフォ管理部１３により、資源開放要求処理が行われ（Ｓ１０６）、資源獲得要求であればセマフォ管理部１３により、資源獲得要求処理処理が行われる（Ｓ１０７）。また、システム呼び出し時の命令がメール送信命令であれば、メール管理部１４により、メール送信処理が行われ（Ｓ１０８）、メールの受信命令であればメール管理部１４により、メール受信処理が行われる（Ｓ１０９）。

【００４４】図３は、イベントフラグ管理部１２が行うフラグ設定命令処理の詳細な処理内容を示すフローチャートである。イベントフラグ管理部１２は、イベントフラグの設定命令において指定されたフラグＩＤにより示されたＯＳのイベントフラグ領域に、指定されたビットパタンを設定し（Ｓ２０１）、当該フラグＩＤを指定してＷａｉｔ状態となっているタスクを検索する（Ｓ２０２）。検索の結果、Ｗａｉｔ状態となっているタスクが存在していれば（Ｓ２０３：Ｙｅｓ）、設定命令により設定されたビットパタンが、Ｗａｉｔ状態のタスクによって指定されているビットパタンと一致するか否かを調べる（Ｓ２０４）。ビットパタンが一致していれば、Ｗａｉｔ状態のタスクが起床することによりシステム状態が変化するため、イベントフラグ管理部１２は、ビットパタンのクリアを行い（Ｓ２０５）、イベントフラグの設定という事象の発生をスケジューリング管理部１７に通知する（Ｓ２０６）。なお、ステップＳ２０５におけるビットパタンのクリアについては、ＯＳの仕様や、処理内容等によっては行わない場合もある。

【００４５】Ｗａｉｔ状態となっているタスクが存在しない（Ｓ２０３：Ｎｏ）か、タスクが存在してもビットパタンが一致しない場合（Ｓ２０４：Ｎｏ）は、システム状態に何ら変化がないことを意味するので、そのままイベントフラグ管理部１２の処理を終了する。図４は、イベントフラグ管理部１２が行うフラグ待ち命令処理の詳細な処理内容を示すフローチャートである。イベントフラグ管理部１２は、フラグの待ち命令において指定されたフラグＩＤを有するＯＳ領域に保持されているビットパタンが、待ち命令により指定されたビットパタンと一致しているか否かを比較する（Ｓ２１１）。一致していれば、システム状態に何ら変化が起きないことを意味するのでイベントフラグ管理部１２は、ビットパタンのクリアを行い（Ｓ２１４）、処理を終了する。ビットパタンのクリア処理を行わない場合もあることはフラグ設定命令処理と同様である。

【００４６】一致していなければ、待ち命令を発行したタスクがＷａｉｔ状態となるので、イベントフラグ管理部１２は、タスクのＷａｉｔ状態への変化という事象の発生をスケジューリング管理部１７に通知する（Ｓ２１３）。図５は、セマフォ管理部１３が行う資源開放要求処理の詳細な処理内容を示すフローチャートである。セマフォ管理部１３は、資源開放命令において指定されたＩＤにより示された資源の割当可能量を保持しているカウンタに、開放された資源の量を加算し（Ｓ３０１）、当該資源の獲得待ちとしてＷａｉｔ状態になっているタスクが存在するか否かを検索する（Ｓ３０２）。検索の結果、Ｗａｉｔ状態となっているタスクが存在する場合にはステップＳ３０１におけるカウンタの加算により起床するタスクが発生するか否かを判定する（Ｓ３０４）。起床するタスクが発生すれば、当該起床するタスクの資源獲得要求量だけ、カウンタの減算を行い（Ｓ３０５）、資源の開放という事象の発生をスケジューリング管理部１７に通知する（Ｓ３０６）。

【００４７】Ｗａｉｔ状態となっているタスクが存在しない場合（Ｓ３０３：Ｎｏ）か、Ｗａｉｔ状態のタスクが存在しても、当該タスクによる資源の獲得要求量が更新後のカウンタの値を上回っている場合（Ｓ３０４：Ｎｏ）には、システム状態に何ら変化がないことを意味するので、そのままセマフォ管理部１３の処理を終了する。

【００４８】図６は、セマフォ管理部１３が行う資源獲得要求処理の詳細な処理内容を示すフローチャートである。セマフォ管理部１３は、資源獲得要求の命令中で指定されたＩＤにより示された資源の割当可能量を保持しているカウンタに保持されている値を、命令中で指定された資源の獲得要求量と比較する（Ｓ３１１）。カウンタに保持されている値が資源の獲得要求量以上である場合には、当該タスクが要求した資源を確保して、そのまま処理の実行を継続することを意味する。従って、セマフォ管理部１３はカウンタの値から資源の要求量を減算して（Ｓ３１４）、処理を終了する。

【００４９】カウンタに保持されている値が資源の獲得要求量を下回っている場合には、当該タスクは資源待ちとしてＷａｉｔ状態となるため、セマフォ管理部１３は、タスクのＷａｉｔ状態への変化という事象の発生をスケジューリング管理部１７に通知する（Ｓ３１３）。図７は、メール管理部１４が行うメール送信処理の詳細な処理内容を示すフローチャートである。

【００５０】メール管理部１４は、命令中で指定されたＩＤにより示されたメールボックスにメッセージを送信する（Ｓ４０１）。その後、当該メールボックスへのメールの送信を待ってＷａｉｔ状態となっているタスクが存在するか否かを検索する（Ｓ４０２）。Ｗａｉｔ状態のタスクが存在する場合は、当該Ｗａｉｔ状態のタスクが起床することを意味する。この場合、メール管理部１４は、当該Ｗａｉｔ状態のタスクにメールを渡し（Ｓ４０４）、メールの受信という事象の発生をスケジューリング管理部１７に通知する（Ｓ４０５）。Ｗａｉｔ状態

のタスクが存在しなければ（Ｓ４０３：Ｎｏ）、システム状態に何ら変化がないことを意味するので、そのままメール管理部１４の処理を終了する。

【００５１】図８は、メール管理部１４が行うメール受信処理の詳細な処理内容を示すフローチャートである。メール管理部１４は、命令中で指定されたメールボックスＩＤを有するメールボックスの内容を検索する（Ｓ４１１）。指定されたメールボックスにメールが存在すれば、当該タスクがそのメールを受信し、そのまま処理を継続実行できる事を意味するので、メール管理部１４は、呼び出したタスクにメールを渡し（Ｓ４１４）、そのまま処理を終了する。

【００５２】指定されたメールボックスにメールが存在しなければ（Ｓ４１２：Ｎｏ）、受信命令を発行したタスクは、他のタスクからのメールの送信があるまでＷａｉｔ状態となるため、メール管理部１４は、タスクのＷａｉｔ状態への変化という事象の発生をスケジューリング管理部１７に通知する（Ｓ４１３）。以上のように、イベントフラグ管理部１２、セマフォ管理部１３、メール管理部１４の何れかによる処理が終了すると、図２のフローチャートに戻って、各事象管理部からのスケジューリング管理部１７への事象の発生の通知が行われたか否かを見る（Ｓ１１０）。ここで、事象の発生の通知が行われていないということは、システム状態に何ら変化がないことを意味するので、スケジューリング管理部１７及びタスク切替え管理部１８の処理は行われず、そのままステップＳ１１１へと進む（Ｓ１１０：Ｎｏ）。

【００５３】各事象管理部から、スケジューリング管理部１７への通知があった場合には、スケジューリング管理部１７の処理が行われる（Ｓ１１１）。図９はスケジューリング管理部１７の詳細な処理内容を示すフローチャートである。スケジューリング管理部１７は、まずイベントフラグ管理部１２、セマフォ管理部１３、メール管理部１４の何れかからの通知を受けて事象の発生を検知し（Ｓ５０１）、状態遷移テーブル保持部１６を参照する。本実施の形態においては、スケジューリング管理部１７は現在のシステム状態を表す状態番号（システム状態には、後述するように状態番号が割り振られている）を保持しているので、現在の状態番号と、発生した事象とから、状態遷移後の状態番号を判定する（Ｓ５０２）。

【００５４】図１５は状態遷移テーブルのデータ構造を示す図である。状態遷移テーブルの生成方法については、後で詳細に説明するが、ここでは、状態遷移テーブルの使用方法について簡単に説明する。図１５に示す状態遷移テーブルは、縦軸に現在の状態番号をとり、横軸に発生した事象をとる。状態番号とは、システム状態ごとに割り振られた番号である。また、この例では横軸として、Ｒｕｎ状態のタスクがＷａｉｔ状態となる場合（以下「Ｗａｉｔ状態への変化」と呼ぶ）、フラグの設定命令が発行され、フラグＩＤが「ＦＬＡＧ＿Ａ」であるイベントフラグ領域にビットパタンが設定された場合（以下「フラグＡの設定」と呼ぶ）、メールボックスＩＤが「ＭＢＸ＿Ｂ」であるメールボックスにメールが送信された場合（以下「メッセージＢの送信」と呼ぶ）、メールボックスＩＤが「ＭＢＸ＿Ｃ」であるメールボックスにメールが送信された場合（以下「メッセージＣの送信」と呼ぶ）の四つの事象が取られており、現在の状態番号により示されるシステム状態において、それぞれの事象が発生した場合の、状態遷移後の状態番号が示されている。

【００５５】即ち、図１５の状態遷移テーブルでは、例えば状態番号１のシステム状態において、マルチタスク制御装置を駆動したタスクがＷａｉｔ状態となった場合には、状態番号４のシステム状態に遷移することを意味している。ここで、状態番号は、優先度の高いタスクがＲｕｎ状態であるシステム状態ほど小さい状態番号を有するように割り振られている。図１５の例では、状態番号が１、２、３であるシステム状態では、最も優先度の高いタスクＡがＲｕｎ状態であり、状態番号が４、５であるシステム状態では、次に優先度の高いタスクＢがＲｕｎ状態であり、状態番号が６であるシステム状態では、最も優先度の低いタスクＣがＲｕｎ状態である。このように状態番号を割り振ることにより、状態番号から、Ｒｕｎ状態となるタスクを判別することが容易になる。

【００５６】スケジューリング管理部１７は、まず、タスク状態データ保持部１５に保持されているタスク状態データを参照して、現在Ｒｕｎ状態であるタスク、即ちマルチタスク制御装置を駆動したタスクを判定する。次に、状態遷移テーブルを参照して得られる状態遷移後のシステム状態を表す状態番号から、状態遷移後にＲｕｎ状態となるタスクを判定し、両者を比較する（Ｓ５０３）ことによりタスク切替えが必要か否かを判定する（Ｓ５０４）。

【００５７】ここで、呼び出し元が割り込みルーチンである場合にはタスク切替えはできないので、タスク切替えは必要でないと判定する（Ｓ５０５）。以上の判定により、タスク切替えが必要である場合にはタスク切替え管理部１８にタスクの切替えを要求して（Ｓ５０６）処理を終了する。なお、呼び出し元が割り込みルーチンである場合（Ｓ５０５：Ｙｅｓ）には、割り込みルーチンに復帰する際にタスク切替えを行うことはできないが、割り込みルーチンによる処理が終了して、タスクに制御が戻るときにタスクの切替えを行う必要がある。そのため、タスク切替えのマーク付けという処理を行う（Ｓ５０７）。タスク切替えのマーク付けとは、ステップＳ５０４における状態遷移後のタスクについて、後のタイミング（タスクに制御が戻るとき）でのタスク切替えを行うために、当該状態遷移後に行うべきタスクの識別子等

を保存する処理である。具体的には、ＯＳ領域に設けられている所定の領域に、切替えるべきタスクを表す識別子を保存する等の方法で行う。

【００５８】スケジューリング管理部１７の処理が終了した後は、図２のフローチャートに戻って、タスク切替えが必要ない場合（Ｓ１１２：Ｎｏ）には、タスク切替え管理部１８による処理は行われずにそのままステップＳ１１４へと進む。タスク切替えが必要な場合は、タスク切替え管理部１８により、レジスタの退避／復帰等の処理及びタスク状態データ保持部１５の更新処理が行われ（Ｓ１１３）、後処理部１９にて、呼び出し元を表す情報の更新、削除等の共通処理が行われて（Ｓ１１４）、マルチタスク制御装置の処理が終了する。

【００５９】以上のように、本実施の形態のマルチタスク制御装置では、事象の発生が検知された場合に、キュー操作やタスク間の優先度の比較等の処理を行わず、状態遷移テーブルを参照することにより状態遷移後のシステム状態を判定するという方法をとっているので、スケジューリング処理を高速に行うことが可能となる。次に、状態遷移テーブルの生成方法について説明する。

【００６０】図１０は、状態遷移テーブル生成部２０の詳細な構成を示すブロック図である。同図に示されるように、状態遷移テーブル生成部２０は、状態生成部２１、状態抽出部２２、テーブル生成部２３から構成される。なお、同図に示されるシステム定義テーブル保持部３０は、システム定義テーブルを保持している部分であり、従来のマルチタスク制御装置においても存在していたものである。最も、システム定義テーブル保持部３０が保持するテーブルの内容については、ＯＳの仕様等によって異なる場合もある。

【００６１】図１１はシステム定義テーブルの内容の一例を示す図である。同図は、種々存在するシステム定義テーブルの中で、状態遷移テーブルの生成に必要な部分を表したものである。同図に示されるように、システム定義テーブルの中には図１１（ａ）に示すタスク定義テーブルと、図１１（ｂ）に示す同期通信機能定義テーブルとが存在する。図１１（ａ）のタスク定義テーブルは、タスク名、タスクの優先度、スタックサイズ等を定義する。同図の優先度はその数字が小さいほど、優先度が高いことを意味している。本実施の形態の例では、タスクＡ、タスクＢ、タスクＣの三つのタスクを定義しているが、タスクの数に特に制限はなく、また複数のタスクに同一の優先度を付与することも可能である。図１１（ｂ）の同期通信機能定義テーブルには、発生し得る事象に関連するフラグＩＤ、資源のＩＤ、メールボックスＩＤと、当該事象と関連のあるタスク名とが定義されている。同図の例では、フラグＩＤが「ＦＬＡＧ＿Ａ」であるイベントフラグがタスクＡと関連している旨の定義がなされているが、これは、「ＦＬＡＧ＿Ａ」というフラグＩＤを有するイベントフラグの設定が事象の発生として検知され、当該事象の発生によりタスクＡが起床することを意味している。

【００６２】メールボックスＩＤ「ＭＢＸ＿Ｂ」、「ＭＢＸ＿Ｃ」についても同様であり、当該メールボックスへのメールの送信が事象の発生として検知され、当該事象の発生によりタスクＢ、タスクＣがそれぞれ起床することを意味している。状態生成部２１は、システム定義テーブル保持部３０に保持されているシステム定義テーブルを参照して、システム状態テーブルを生成する。

【００６３】図１２は、システム状態テーブルのデータ構造を示す図である。同図に示されるように、システム状態テーブルは、システム定義テーブルに定義されているタスクが取り得る状態の組み合わせを全て定義したものである。同図の例では三つのタスクが定義されており、各々の優先度は全て異なっているが、複数のタスクの優先度が同一である場合も有り得る。そのような場合は、同一の優先度のタスクについては、起床した順番によりシステム状態が異なるものとみなしてシステム状態テーブルを生成する。

【００６４】状態抽出部２２は、ＯＳ規則に基づく状態抽出部２２１と、アプリケーション固有規則に基づく状態抽出部２２２とを含む。例えば、図１３（ａ）に示すようなＯＳの機能に基づくシステム状態に関する規則（以下「ＯＳ規則」と呼ぶ）及び図１３（ｂ）に示すようなアプリケーション固有の理由に基づくシステム状態に関する規則（以下「アプリケーション固有規則」と呼ぶ）が存在する場合に、ＯＳ規則に基づく状態抽出部２２１は、ＯＳ規則に合致するシステム状態のみをシステム状態テーブルから抽出し、アプリケーション固有規則に基づく状態抽出部２２２は、アプリケーション固有規則に合致するシステム状態のみをさらに抽出する。状態抽出部２２は、抽出されたシステム状態に状態番号を割り振り直すことにより、システム状態最小テーブルを生成する。図１４（ｂ）は、システム状態最小テーブルの一例である。

【００６５】テーブル生成部２３は、システム状態最小テーブルから状態遷移テーブルを生成する。次に、状態遷移テーブル生成部２０の処理内容について説明する。図１６は状態遷移テーブル生成部２０の処理内容を示すフローチャートである。状態遷移テーブル生成部２０はまず、状態生成部２１による処理を行う（Ｓ７０１）。状態生成部２１は、前述のとおり、図１１に示すシステム定義テーブルを参照して、図１２に示すシステム状態テーブルを生成する。具体的には、図１１（ａ）に示されるタスク定義テーブルからタスク名を抽出し、図１１（ｂ）に示される同期通信機能定義テーブルから関連する事象を抽出して、各々のタスクが取り得る状態の組み合わせを全て定義することにより、システム状態テーブルを生成する。ここで、Ｗａｉｔ（Ｆａ）は、フラグＩＤ「ＦＬＡＧ＿Ａ」を有するイベントフラグの設定を待

つＷａｉｔ状態であり、Ｗａｉｔ（Ｍｂ）は、メールボックスＩＤ「ＭＢＸ__Ｂ」を有するメールボックスへのメールの送信を待つＷａｉｔ状態であることを意味する。Ｗａｉｔ（Ｍｃ）についてもＷａｉｔ（Ｍｂ）と同様である。

【００６６】システム状態テーブルの生成を終了すると、図１６のフローチャートに戻って、状態遷移テーブル生成部２０は、状態抽出部２２による処理として、まず、ＯＳ規則に基づく状態抽出部２２１による処理を行う（Ｓ７０２）。図１７はＯＳ規則に基づく状態抽出処理の詳細な処理内容を示すフローチャートである。ＯＳ規則に基づく状態抽出部２２１は、まず図１３（ａ）に示すＯＳ規則（１）に基づいて、システム状態テーブルから、Ｒｕｎ状態のタスクがシステムで二つ以上存在する組み合わせを削除する（Ｓ８０１）。この処理により、システム状態テーブルから、状態番号が５、６、８、９、１１～１８、２０～２７であるシステム状態を表す組み合わせが抽出される。

【００６７】次に、ＯＳ規則に基づく状態抽出部２２１は、ＯＳ規則（２）に基づいて、最高優先度のタスクがＲｅａｄｙ状態となっている組み合わせを削除する（Ｓ８０２）。本実施の形態の例では、最高優先度のタスクはタスクＡであるので、この処理により、状態番号が５、６、８、９、２０～２７であるシステム状態を表す組み合わせが抽出される。

【００６８】次に、ＯＳ規則に基づく状態抽出部２２１は、ＯＳ規則（３）に基づいて、Ｒｕｎ状態のタスクが存在する場合に、そのタスクよりも優先度の高いタスクがＲｅａｄｙ状態となっている組み合わせを削除する（Ｓ８０３）。本実施の形態の例では、タスクＡ、タスクＢ、タスクＣの順に優先度が高いので、ステップＳ８０２における抽出の結果を考慮すると、本ステップの処理で削除の対象となるのはタスクＢがＲｅａｄｙ状態であり、かつ、タスクＣがＲｕｎ状態である組み合わせである。即ち、この処理により、状態番号が５、６、８、９、２０、２１、２３～２７であるシステム状態を表す組み合わせが抽出される。

【００６９】最後に、ＯＳ規則に基づく状態抽出部２２１は、ＯＳ規則（４）に基づいてＲｅａｄｙ状態のタスクが存在し、かつ、Ｒｕｎ状態のタスクが存在しない組み合わせを削除する（Ｓ８０４）。この処理により、状態番号が５、６、８、９、２０、２１、２５、２７であるシステム状態を表す組み合わせが抽出される。ＯＳ規則に基づく状態抽出処理を終了すると、図１６のフローチャートに戻って状態抽出部２２は、アプリケーション固有規則に基づく状態抽出処理を行う（Ｓ７０３）。アプリケーション固有規則に基づく状態抽出部２２２は、図１３（ｂ）に示すアプリケーション固有規則に基づいて、ＯＳ規則に基づく状態抽出処理で抽出されたシステム状態から、タスクＢとタスクＣとが同時にＷａｉｔ状態となる組み合わせを削除する。この処理により、状態番号が５、６、８、２０、２１、２５であるシステム状態を表す組み合わせが抽出される。

【００７０】以上の抽出処理により、状態抽出部２２は、図１４（ａ）に示すような状態抽出後のシステム状態テーブルを得る。状態抽出部２２は、テーブルのサイズ効率等の点で有利なように状態番号を割り当て直し、最終的に図１４（ｂ）に示すようなシステム状態最小テーブルを生成する。前述の如く、本実施の形態では、状態番号の割り当ては、優先度の高いタスクがＲｕｎ状態であるシステム状態ほど小さい状態番号を有するように行っているが、状態番号の順番については、逆に、優先度の高いタスクがＲｕｎ状態であるシステム状態ほど大きい状態番号を有するように行うことも可能である。何れの場合も、状態番号の範囲からＲｕｎ状態となるタスクを判別することができる。

【００７１】状態抽出部２２によるシステム状態最小テーブルの生成が終了すると、テーブル生成部２３が、システム状態最小テーブルと、システム定義テーブルとを参照して、状態遷移テーブルを生成する（Ｓ７０４）。状態遷移テーブルとは、前述のとおり、現在のシステム状態と、発生した事象とから、状態遷移後のシステム状態を判定すべく定義されたテーブルである。

【００７２】図１８及び図１９は、テーブル生成部２３の詳細な処理内容を示すフローチャートである。テーブル生成部２３は、まず、システム定義テーブルを参照して、タスク及びタスクに関連した事象の情報を抽出する（Ｓ９０１）。例えば図１１に示した例では、タスク定義テーブルから切替えが行われるタスク及びタスクの優先度が抽出され、同期通信機能定義テーブルに定義されているタスクに関連したフラグＩＤ、メールボックスＩＤ等の情報からはタスクに関連した事象が抽出される。ここで、抽出とは必ずしも別の領域に抽出することを意味せず、テーブルの生成中に必要に応じてシステム定義テーブルを参照しても差し支えない。

【００７３】ループＡの処理は、システム状態最小テーブルに含まれる全てのシステム状態について順次行われる（Ｓ９０２）。テーブル生成部２３は、システム状態最小テーブルから一つずつシステム状態を読み出す（Ｓ９０３）。ループＢの処理は、Ｒｕｎ状態のタスクがＷａｉｔ状態に変化する場合と、ステップＳ９０１で同期通信機能定義テーブルから抽出された事象が発生した場合とを含む、発生し得る全ての事象について行われる（Ｓ９０４）。

【００７４】即ち、ループＡ及びループＢの処理は、ステップＳ９０３において読み出されたシステム状態を現在のシステム状態と仮定し、当該システム状態において事象が発生した場合の状態遷移を、発生した事象ごとにシミュレートして、図１５のような状態遷移テーブルのマトリックスを一つずつ埋めていく処理である。以下、

ループBにおける処理の内容について、具体例を示しながら詳細に説明する。

【0075】テーブル生成部23は、まず、どの事象が発生した場合についてシミュレートを行うかを決定する（S905）。その後、ステップS905において決定された事象が、Run状態のタスクがWait状態に変化する場合であるか否かを判定する（S906）。これは、Run状態のタスクがWait状態に変化する場合と、事象の発生によりタスクが起床した場合とでは、明らかに異なる種類の状態遷移を伴うからである。

【0076】Run状態のタスクがWait状態に変化する場合についての状態遷移のシミュレート処理であれば（S906：Yes）、まず、選択されたシステム状態においてRun状態であったタスクが、Wait状態に変化するものと仮定する（S907）。例えば、状態番号1のシステム状態において、Run状態のタスクがWait状態に変化した場合を仮定すれば、図20に示す例のように、まずタスクAがWait状態に変化したものとする（図20（b））。

【0077】次に、残りのタスクの中で、Ready状態となっているタスクがあるか否かを判定する（S908）。Ready状態のタスクがなければ、図19のフローチャートに移って、状態遷移後のシステム状態の状態番号を検索する（S909）。ここで、該当するシステム状態がシステム状態最小テーブルに存在すれば（S910：Yes）、該当するシステム状態の状態番号を状態遷移テーブルの対応する位置に保存する（S911）。しかし、前記システム状態がシステム状態最小テーブルに存在しない場合は、状態遷移後のシステム状態を未定義とする（S912）。この場合、状態遷移テーブルのマトリックスには、状態遷移後の状態番号が未定義であることを示すフラグ（図15の例では「－」）が設定される。

【0078】ステップS908において、Ready状態のタスクが存在すれば、図19のフローチャートに移って、Ready状態のタスクの中で最も優先度が高いタスクが複数存在するか否かを判定する（S913）。図20の例であれば、図20（b）の状態においてReady状態のタスクの中で最も優先度の高いタスクはタスクBのみである。従ってステップS913ではNoと判定されることになる。

【0079】最も優先度の高いReady状態のタスクが一つであれば（S913：No）、事象の発生により、そのタスクがRun状態となるため、Ready状態のタスクをRun状態として状態遷移後のシステム状態を決定する（S914）。図20の例では、タスクBがRun状態となる（図20（c））。最も優先度の高いReady状態のタスクが複数存在する場合は、どのタスクをRun状態とするかをシステムの仕様等に基づいて決定しておく必要がある。例えば最先に起床したタスクをRun状態にする方法が考えられるが、他の方法で決定してもよい。何れにしても、状況に応じて適切なタスクがRun状態に変化することになる（S915）。ここでの詳細な説明は割愛するが、優先度が同一であるタスクについては、システム状態テーブル生成時に起床した順番ごとに別のシステム状態として定義することにより対処を行う。

【0080】以上のように状態遷移後のシステム状態が決定されると、決定されたシステム状態に対応する状態番号をシステム状態最小テーブルから検索し（S909）、該当するシステム状態が、システム状態最小テーブルに存在すれば（S910：Yes）、検索された状態番号を状態遷移テーブルに保存する（S911）。図20の例であれば、図20（c）に示されるシステム状態の状態番号は4であるため、図20（d）に示されるように、状態番号4が状態遷移テーブルの所定の位置に保存されることになる。ステップS910において、システム状態がシステム状態最小テーブルに存在しなければ、状態遷移後のシステム状態を未定義とする（S912）。

【0081】一方、ステップS906において、Noと判定された場合、即ち、Run状態のタスクがWait状態となる場合以外の事象の発生を仮定する場合には、ステップS905で決定された事象に関連するタスクが、現在のシステム状態においてWait状態であるか否かを判定する（S916）。Wait状態でない場合とは、システム状態に何ら変更が無いことを意味するものであり、状態遷移テーブルが参照されない場合であるので、状態遷移後のシステム状態を未定義とし（S912）、ループBの先頭に戻る。Wait状態である場合は、事象の発生によりタスクが起床することとなるので、当該Wait状態のタスクをReady状態とする（S917）。

【0082】事象の発生によるシステム状態の遷移をシミュレートする処理の一例を図21に示す。同図に示される例は、状態番号6のシステム状態において、イベントフラグID「FLAG_A」を有するイベントフラグのOS領域に、タスクCが発行したフラグ設定命令によりビットパタンの設定がなされることによりタスクAが起床する場合の例である。同図に示される例では、「FLAG_A」のビットパタンの一致により、タスクAがReady状態となる（図21（b））。

【0083】次に、テーブル生成部23はステップS917でReady状態となったタスクの優先度と、現在のシステム状態においてRun状態であるタスクの優先度とを比較する（S918）。Run状態のタスクの優先度の方が、Ready状態のタスクの優先度より高い場合（S918：Yes）には、Run状態のタスクがそのまま継続実行されるので、図19のフローチャートに移り、状態遷移後のシステム状態を表す状態番号を検

索し（Ｓ９０９）、システム状態がシステム状態最小テーブルに存在すれば（Ｓ９１０：Ｙｅｓ）、状態番号を状態遷移テーブルのマトリックスの所定の位置に保存する（Ｓ９１１）。

【００８４】図２１（ｂ）に示す例のように、Ｒｅａｄｙ状態のタスクの優先度の方が、Ｒｕｎ状態のタスクの優先度より高い場合（Ｓ９１８：Ｎｏ）には、Ｒｕｎ状態のタスクをＲｅａｄｙ状態に変更し（Ｓ９１９）、図１９のフローチャートのステップＳ９１３に進む。図２１（ｂ）の例ではステップＳ９１９においてタスクＣがＲｅａｄｙ状態に変更され（図２１（ｃ））、先に説明したステップＳ９１３の判定の後、ステップＳ９１４においてタスクＡをＲｕｎ状態とする（図２１（ｄ））。

【００８５】以上のように状態遷移後のシステム状態が決定されると、ステップＳ９０９において、決定されたシステム状態に対応する状態番号をシステム状態最小テーブルから検索し、システム状態がシステム状態最小テーブルに存在すれば（Ｓ９１０：Ｙｅｓ）、検索された状態番号を状態遷移テーブルに保存する（Ｓ９１１）。図２１の例であれば、図２１（ｄ）に示されるシステム状態の状態番号が３であるため、図２１（ｅ）に示されるように、状態番号３が状態遷移テーブルの所定の位置に保存されることになる。

【００８６】以上のような処理を発生し得る全ての事象について行い（Ｓ９２０）、さらに、システム状態最小テーブルに含まれている全ての状態について行う（Ｓ９２１）ことで、図１５に示すような状態遷移テーブルを機械的に生成することができる。図１５に示す状態遷移テーブルにおいて、網掛け部分の状態遷移においてタスク切替えが発生することを意味している。

【００８７】なお、本実施の形態では、最も一般的な例として、各タスクが取り得る状態がＷａｉｔ状態、Ｒｅａｄｙ状態、Ｒｕｎ状態の三種類であるシステムについて詳細に説明したが、ＯＳの機能によっては、他の状態を取り得る場合もある。例えばサスペンド状態、サスペンド待機状態、休止状態等である。図２２に、それらの状態を含めた状態遷移図を示す。ここで、前記各状態について簡単に説明しておくと、サスペンド状態とは、タスクが強制的な待ち状態に置かれている状態である。具体的には、当該タスクがＲｅａｄｙ状態である場合において他のタスクから中断命令が発行された場合や、サスペンド待機状態において、フラグの設定命令等により待ちが解除された場合にとり得る状態であり、他のタスクから、再開命令が発行されることによりＲｅａｄｙ状態となる。ここで、中断命令とは、例えば"suspend(taskID)"というようにタスクＩＤを指定して発行され、再開命令も、例えば"resume(taskID)"というようにタスクＩＤを指定して発行されるが、何れもシステム状態の変化を伴う。

【００８８】サスペンド待機状態とは、Ｗａｉｔ状態において中断命令が発行された場合にとり得る状態であり、前述の如く、事象の発生による待ちの解除によりサスペンド状態に遷移する。また、他のタスクから再開命令が発行された場合には、Ｗａｉｔ状態に戻る。休止状態とは、スタック等のタスク実行のための最小限の資源をも開放し、タスクの実行を休止している状態である。休止状態へは、Ｒｕｎ状態のタスク自身の中で終了命令を発行した場合や、Ｒｅａｄｙ状態、Ｗａｉｔ状態等の状態から強制終了命令が発行された場合に遷移する。また、休止状態からは、他のタスクから起動命令が発行された場合に、Ｒｅａｄｙ状態に移行する。

【００８９】ここで、終了命令は、例えば"exittask"というような命令をＲｕｎ状態のタスク自体が発行することにより、当該Ｒｕｎ状態のタスク自身を休止状態に遷移させる。中断命令は、例えば"terminate(taskID)"というようにタスクＩＤを指定して発行され、起動命令は、例えば"starttask(taskID)"というようにタスクＩＤを指定して発行される。何れもシステム状態の変化を伴う。

【００９０】しかしながら、図２２に示された状態遷移図及び以上の説明から明らかなように、何れの状態についても、マルチタスク制御装置を駆動する際のシステム呼び出し命令に基づく事象の発生により状態遷移が起こる点において前記Ｗａｉｔ、Ｒｅａｄｙ、Ｒｕｎの三種類の状態と本質的に同一であり、状態遷移テーブルを用いたタスク切替えについても、状態遷移テーブルの生成についても、今回詳細に説明した方法を若干拡張することにより容易に適用することが可能である。

【００９１】即ち、タスク切替え時にあっては、前記中断命令や、再開命令等の発行を事象の発生とみなして、状態遷移テーブルを参照することにより状態遷移後のシステム状態を判定し、さらに状態遷移後のシステム状態から状態遷移後に実行すべきタスクを判定する。状態遷移テーブルの生成においては、本実施の形態において説明したような事象発生時の状態遷移のシミュレート処理を、前記中断命令や、再開命令等についても、状態遷移図に基づいて行うようにすれば良い。

【００９２】また、本実施の形態では、状態遷移テーブルとして、状態番号番号ごとに、各事象が発生した後のシステム状態を表す状態番号を保持する形としたが、テーブルのフォーマットは、本実施の形態の例に限られず、状態遷移後に実行すべきタスクが決定できるものであれば他のフォーマットにしてもよく、例えば、システム状態を表すテーブルを直接保持するようにしても構わない。

【００９３】また、本実施の形態では、処理の高速化を優先して、タスク状態データ保持部１５に、現在実行中のタスクに関する情報を保持する形にしたが、前述の如く、スケジューリング管理部１７に保持されている状態番号から、現在実行中のタスクを判別することも可能で

ある。また、本実施の形態ではスケジューリング管理部１７にその時点での状態番号を保持したが、システム状態を認識することができれば、必ずしも状態番号の形で保持する必要はない。また、システム状態を識別するための情報も、常に保持せずに、事象の発生時にシステム状態を検知するようにしてもよい。

【００９４】また、状態抽出部２２による状態抽出処理においても、本実施の形態で用いたシステム状態に関する規則以外のシステム状態に関する規則に基づいて、システム状態最小テーブルを生成することも可能である。また、本実施の形態では、前処理部１１で行われる処理について、その全ての処理を毎回実行するようにしたが、前処理部１１にて行うべき処理の中には、必ずしも毎回行う必要はない処理が含まれている場合もある。そのような処理については、例えば各事象管理部によりシステム状態の変化が検知された場合のみに、各事象管理部の処理の後で行うようにすることも可能である。その場合には、前処理部１１の処理として行われなかった処理に対応する後処理については、後処理部１９においても行う必要はない。

【００９５】

【発明の効果】以上の説明から明らかなように、本発明に係るマルチタスク制御装置は、複数のタスクを実行するシステムに備えられ、事象の発生に応じて、前記システムが実行するタスクの切替えを行うマルチタスク制御装置において、前記複数のタスクのそれぞれが、実行状態、実行可能状態、待機状態を含む各種の状態の中のどの状態にあるかを表すシステム状態ごとに、各事象が発生した場合に次に実行すべきタスクを決定する情報を保持する状態遷移テーブルと、事象の発生を検知する事象検知手段と、前記事象検知手段により事象の発生が検知された場合に、前記状態遷移テーブルを参照し、その時点におけるシステム状態から次に実行すべきタスクを判定する判定手段と、前記判定手段の判定に応じてタスクの切替えを行うタスク切替え手段とを備え、判定手段は、前記事象検知手段により事象の発生が検知された場合に、システム状態ごとに、各事象が発生した場合に次に実行すべきタスクを決定する情報を保持する状態遷移テーブルを参照して、その時点におけるシステム状態から次に実行すべきタスクを判定するので、事象の発生時に、状態遷移キューのサーチ、キュー接続などのキュー操作や、Ｒｕｎ状態のタスクとのＲｅａｄｙ状態のタスクの優先度の比較処理、Ｒｅａｄｙ状態のタスク同士の優先度の比較処理、タスク切替え時の状態遷移キューの繋ぎかえ操作等を行う必要がなく、スケジューリング処理を高速に行うことが可能になるという効果がある。

【００９６】また、前記判定手段は、その時点におけるシステム状態を保持する保持部を有することも可能である。その場合には、各事象の発生時に、その時点でのシステム状態を検知する必要がないため、スケジューリング処理をより高速に行うことができるという効果がある。また、前記状態遷移テーブルは、前記システムにおいて存在し得るシステム状態について、システム状態ごとに、各事象が発生した場合の遷移先のシステム状態を保持することが可能である。前記状態遷移テーブルが前記システムにおいて存在し得るシステム状態についてのみ保持することにより、状態遷移テーブルを保持する領域が節約できるとともに、システム状態ごとに、各事象が発生した場合の遷移先のシステム状態を保持することにより、状態遷移後のシステム状態を、各々のタスクの状態を調べることなく、迅速に認識することができるという効果がある。

【００９７】また、前記状態遷移テーブルにおいて、各々のシステム状態には、優先度の高いタスクが実行状態であるシステム状態から、昇順又は降順にて状態番号が割り振られ、前記判定手段は、状態遷移テーブルに保持されている遷移先のシステム状態の状態番号から次に実行すべきタスクを判定することが可能である。システム状態の抽出処理後に、システム状態に状態番号を割り振り直すことにより、状態遷移テーブルを保持する領域が節約できるとともに、前記の如く状態番号を割り振っておけば、前記判定手段が、どのタスクがＲｕｎ状態となるかを容易に判定できるという効果がある。

【００９８】また、タスクと、タスクごとの優先度と、各々のタスクを実行可能状態に起床させる事象との定義を含むシステム定義テーブルから、前記状態遷移テーブルを生成する状態遷移テーブル生成手段をさらに備えることもできる。この場合には、状態遷移テーブル生成手段が、システム定義テーブルから機械的に状態遷移テーブルを生成することにより、状態遷移テーブルの生成ミスを防止することができるという効果がある。

【００９９】また、前記状態遷移テーブル生成手段は、前記システム定義テーブルから、各タスクの状態の全ての組み合わせを生成する状態生成手段と、所定のシステム状態に関する規則に従って、前記状態生成手段が生成した組み合わせから、前記システムにおいて存在し得ないシステム状態を表す組み合わせを削除することにより、前記システムにおいて存在し得るシステム状態を表す組み合わせを抽出する状態抽出手段と、前記状態抽出手段により抽出された組み合わせから、前記状態遷移テーブルを生成するテーブル生成手段とを備えることが可能である。

【０１００】状態生成手段が、システム定義テーブルを参照して、各タスクの状態の全ての組み合わせを生成し、状態抽出手段は、前記各タスクの状態の全ての組み合わせから、システムにおいて存在し得ないシステム状態を表す組み合わせを削除することにより、システムにおいて存在し得るシステム状態を表す組み合わせを抽出し、テーブル生成手段は、前記状態抽出手段により抽出された組み合わせから状態遷移テーブルを生成すること

で、システムにおいて存在し得ないシステム状態について状態遷移テーブルを生成することがなくなり、状態遷移テーブルを保持する領域を節約することが可能となる他、簡単なアルゴリズムで、状態遷移テーブルの生成ミスを防止することができるという効果がある。

【０１０１】また、前記状態抽出手段は、前記システムの機能に基づくシステム状態に関する規則と、アプリケーション固有の理由に基づくシステム状態に関する規則とに従って前記システム状態を表す組み合わせを抽出することが可能である。このような処理を行うことにより、オペレーティングシステムの機能の変更や、アプリケーションの変更に柔軟に対応できるという効果を有する。

【図面の簡単な説明】

【図１】本発明の一実施の形態におけるマルチタスク制御装置の構成を示すブロック図である。

【図２】本発明の一実施の形態におけるマルチタスク制御装置の処理内容を示すフローチャートである。

【図３】イベントフラグ管理部が行うフラグ設定命令処理の詳細な処理内容を示すフローチャートである。

【図４】イベントフラグ管理部が行うフラグ待ち命令処理の詳細な処理内容を示すフローチャートである。

【図５】セマフォ管理部が行う資源開放要求処理の詳細な処理内容を示すフローチャートである。

【図６】セマフォ管理部が行う資源獲得要求処理の詳細な処理内容を示すフローチャートである。

【図７】メール管理部が行うメール送信処理の詳細な処理内容を示すフローチャートである。

【図８】メール管理部が行うメール受信処理の詳細な処理内容を示すフローチャートである。

【図９】スケジューリング管理部の詳細な処理内容を示すフローチャートである。

【図１０】状態遷移テーブル生成部の構成を示すブロック図である。

【図１１】（ａ）システム定義テーブルに含まれるタスク定義テーブルの内容の一例を示す図である。

（ｂ）システム定義テーブルに含まれる同期通信機能定義テーブルの内容の一例を示す図である。

【図１２】システム状態テーブルのデータ構造を示す図である。

【図１３】（ａ）ＯＳ規則の一例を示す図である。

（ｂ）アプリケーション固有規則の一例を示す図である。

【図１４】（ａ）状態抽出部による状態抽出後のシステム状態テーブルの一例を示す図である。

（ｂ）システム状態最小テーブルの一例を示す図である。

【図１５】状態遷移テーブルの一例を示す図である。

【図１６】状態遷移テーブル生成部の処理内容を示すフローチャートである。

【図１７】ＯＳ規則に基づく状態抽出処理の処理内容を示すフローチャートである。

【図１８】テーブル生成部の処理内容を示すフローチャートである。

【図１９】テーブル生成部の処理内容を示すフローチャートである。

【図２０】テーブル生成部の処理内容を説明するための図である。

【図２１】テーブル生成部の処理内容を説明するための図である。

【図２２】サスペンド状態、サスペンド待機状態、休止状態を含めた状態遷移図である。

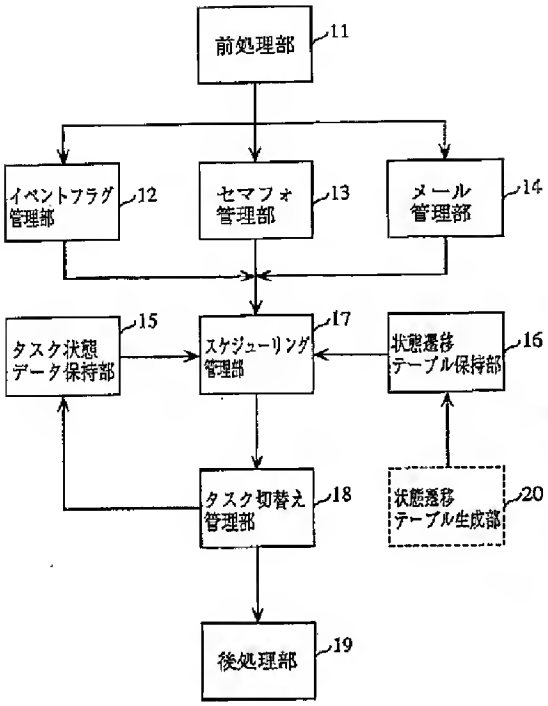
【図２３】従来のマルチタスク制御装置の構成を示すブロック図である。

【図２４】従来のマルチタスク制御装置の動作について説明するための図である。

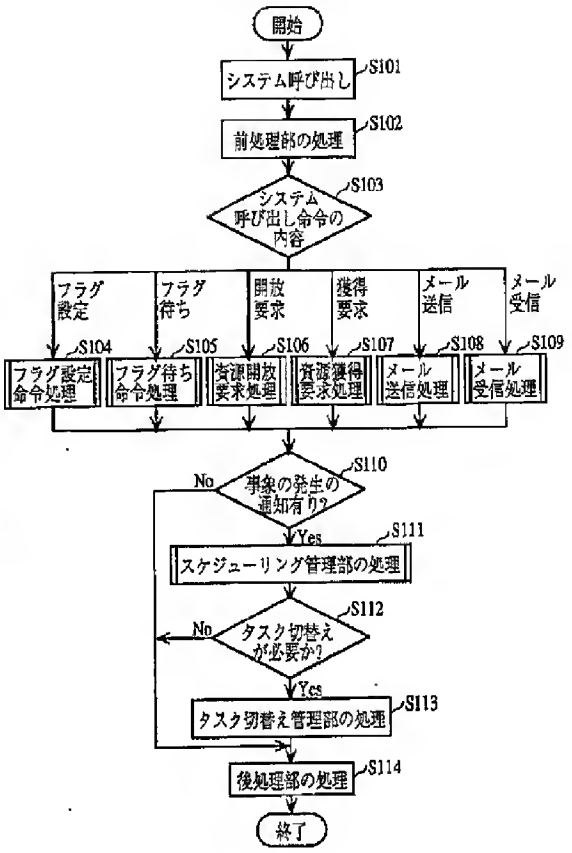
【符号の説明】

１１　前処理部
１２　イベントフラグ管理部
１３　セマフォ管理部
１４　メール管理部
１５　タスク状態データ保持部
１６　状態遷移テーブル保持部
１７　スケジューリング管理部
１８　タスク切替え管理部
１９　後処理部
２０　状態遷移テーブル生成部
２１　状態生成部
２２　状態抽出部
２２１　ＯＳ規則に基づく状態抽出部
２２２　アプリケーション固有規則に基づく状態抽出部
２３　テーブル生成部
３０　システム定義テーブル保持部

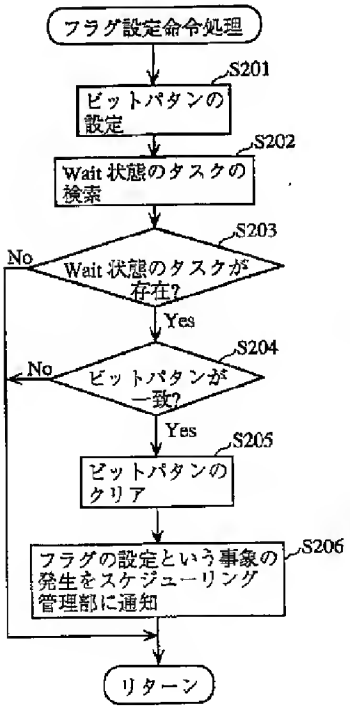
【図1】


前処理部
11
イベントフラグ管理部
12
セマフォ管理部
13
メール管理部
14
タスク状態データ保持部
15
スケジューリング管理部
17
状態遷移テーブル保持部
16
タスク切替え管理部
18
状態遷移テーブル生成部
20
後処理部
19

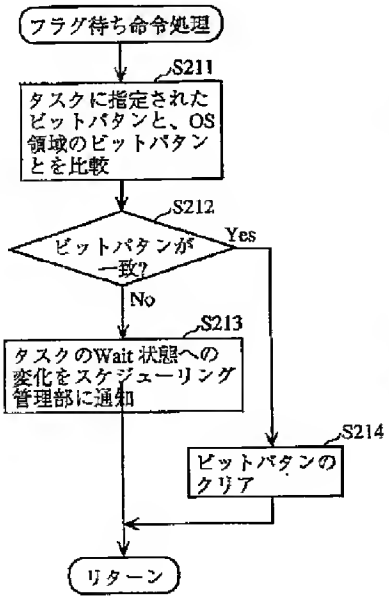
【図2】


開始
システム呼び出し
S101
前処理部の処理
S102
システム呼び出し命令の内容
S103
フラグ設定
フラグ待ち
開放要求
獲得要求
メール送信
メール受信
S104
S105
S106
S107
S108
S109
フラグ設定命令処理
フラグ待ち命令処理
資源開放要求処理
資源獲得要求処理
メール送信処理
メール受信処理
事象の発生の通知有り?
S110
No
Yes
スケジューリング管理部の処理
S111
タスク切替えが必要か?
S112
No
Yes
タスク切替え管理部の処理
S113
後処理部の処理
S114
終了

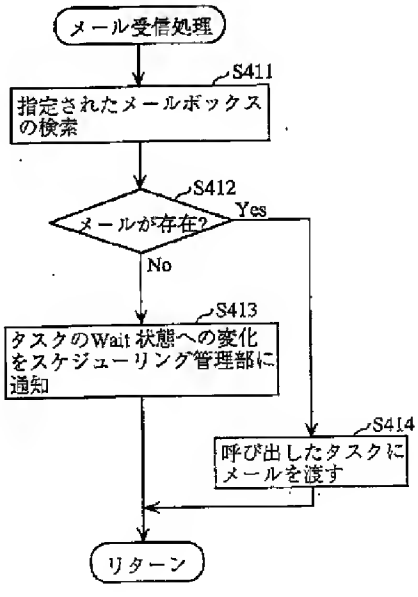
【図3】


フラグ設定命令処理
ビットパタンの設定
S201
Wait状態のタスクの検索
S202
Wait状態のタスクが存在?
S203
No
Yes
ビットパタンが一致?
S204
No
Yes
ビットパタンのクリア
S205
フラグの設定という事象の発生をスケジューリング管理部に通知
S206
リターン

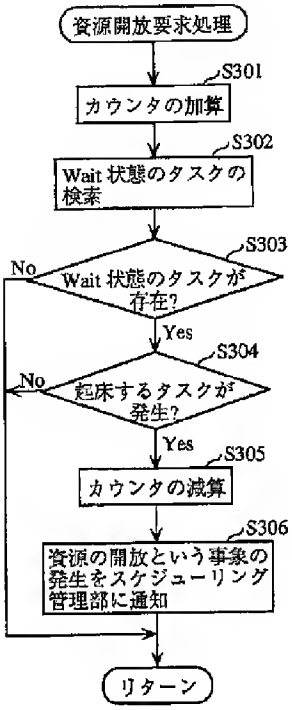
【図4】


フラグ待ち命令処理
タスクに指定されたビットパタンと、OS領域のビットパタンとを比較
S211
ビットパタンが一致?
S212
Yes
No
タスクのWait状態への変化をスケジューリング管理部に通知
S213
ビットパタンのクリア
S214
リターン

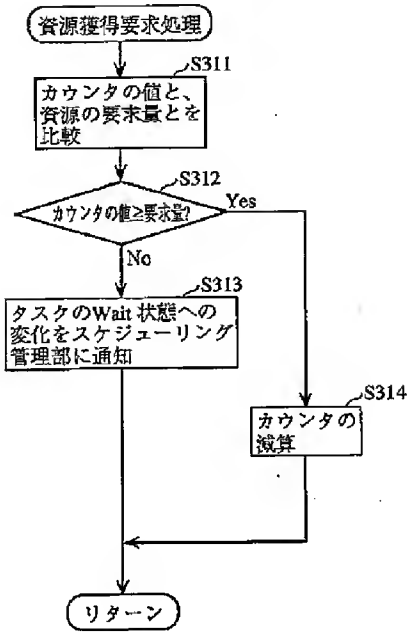
【図8】


メール受信処理
指定されたメールボックスの検索
S411
メールが存在?
S412
Yes
No
タスクのWait状態への変化をスケジューリング管理部に通知
S413
呼び出したタスクにメールを渡す
S414
リターン

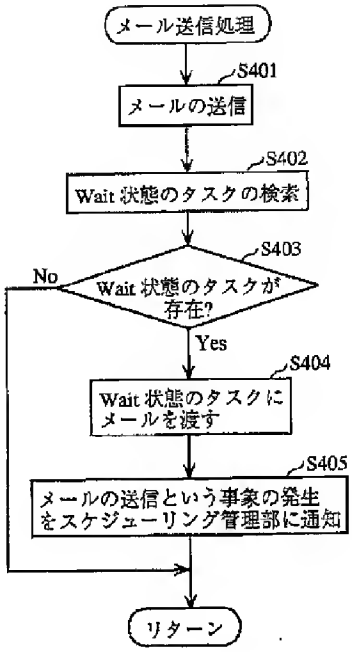
【図5】


資源開放要求処理
S301
カウンタの加算
S302
Wait 状態のタスクの検索
S303
Wait 状態のタスクが存在?
No
Yes
S304
起床するタスクが発生?
No
Yes
S305
カウンタの減算
S306
資源の開放という事象の発生をスケジューリング管理部に通知
リターン

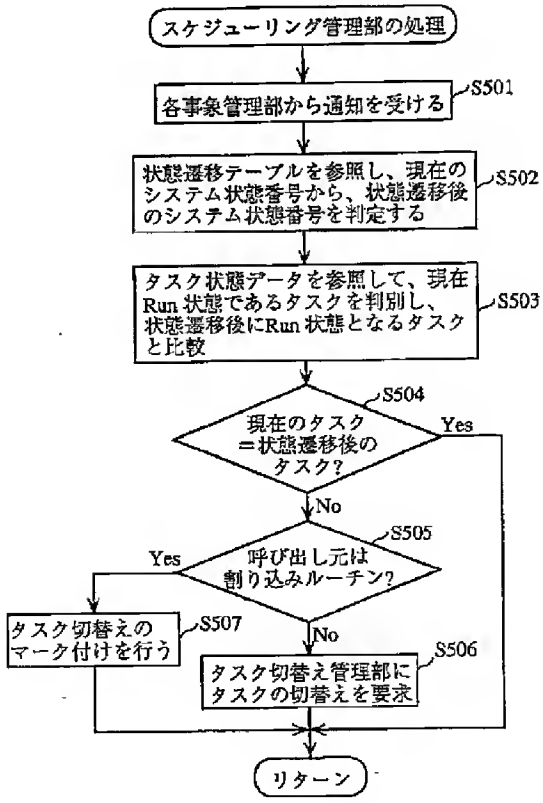
【図6】


資源獲得要求処理
S311
カウンタの値と、資源の要求量とを比較
S312
カウンタの値≥要求量?
Yes
No
S313
タスクのWait 状態への変化をスケジューリング管理部に通知
S314
カウンタの減算
リターン

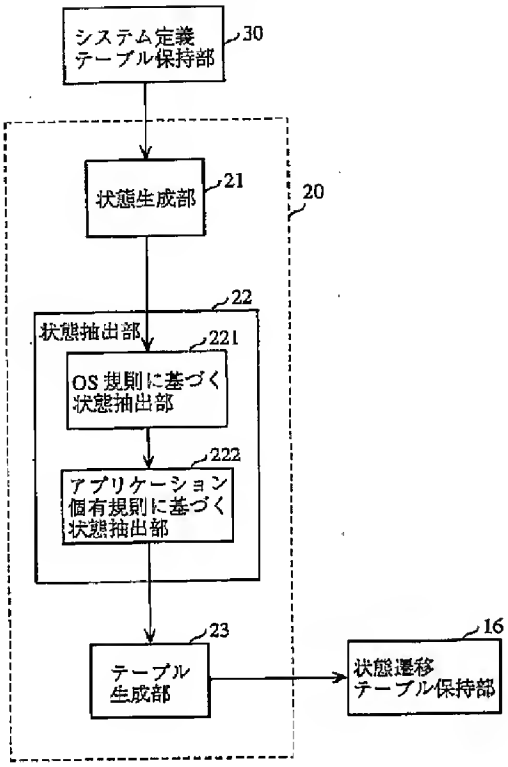
【図7】


メール送信処理
S401
メールの送信
S402
Wait 状態のタスクの検索
S403
Wait 状態のタスクが存在?
No
Yes
S404
Wait 状態のタスクにメールを渡す
S405
メールの送信という事象の発生をスケジューリング管理部に通知
リターン

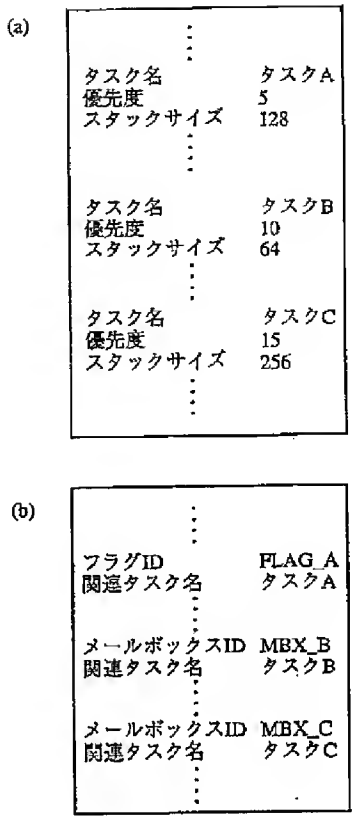
【図9】


スケジューリング管理部の処理
S501
各事象管理部から通知を受ける
S502
状態遷移テーブルを参照し、現在のシステム状態番号から、状態遷移後のシステム状態番号を判定する
S503
タスク状態データを参照して、現在Run 状態であるタスクを判別し、状態遷移後にRun 状態となるタスクと比較
S504
現在のタスク＝状態遷移後のタスク?
Yes
No
S505
呼び出し元は割り込みルーチン?
Yes
No
S507
タスク切替えのマーク付けを行う
S506
タスク切替え管理部にタスクの切替えを要求
リターン

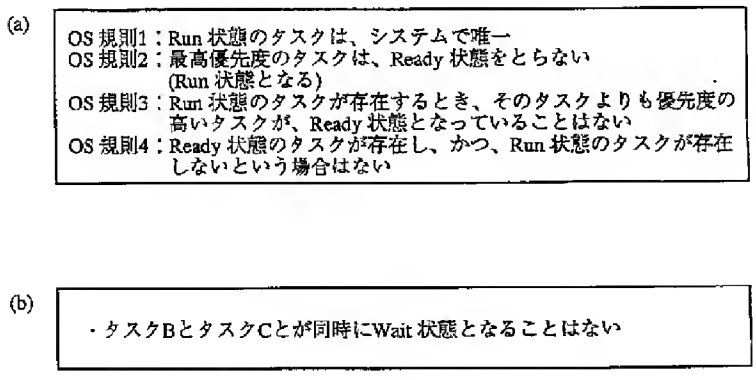
【図１０】



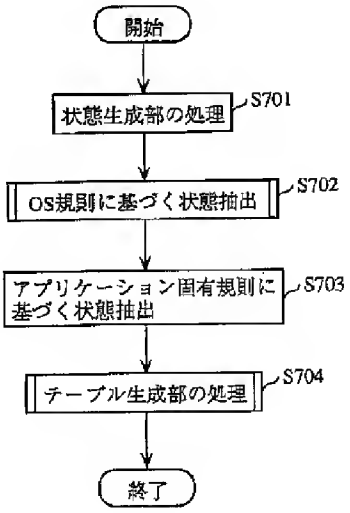
【図１１】



【図１３】



【図１６】



【図１２】

タスク名 / 状態番号	タスクA	タスクB	タスクC
1	Run	Run	Run
2	Run	Run	Ready
3	Run	Run	Wait(Mc)
4	Run	Ready	Run
5	Run	Ready	Ready
6	Run	Ready	Wait(Mc)
7	Run	Wait(Mb)	Run
8	Run	Wait(Mb)	Ready
9	Run	Wait(Mb)	Wait(Mc)
10	Ready	Run	Run
11	Ready	Run	Ready
12	Ready	Run	Wait(Mc)
13	Ready	Ready	Run
14	Ready	Ready	Ready
15	Ready	Ready	Wait(Mc)
16	Ready	Wait(Mb)	Run
17	Ready	Wait(Mb)	Ready
18	Ready	Wait(Mb)	Wait(Mc)
19	Wait(Fa)	Run	Run
20	Wait(Fa)	Run	Ready
21	Wait(Fa)	Run	Wait(Mc)
22	Wait(Fa)	Ready	Run
23	Wait(Fa)	Ready	Ready
24	Wait(Fa)	Ready	Wait(Mc)
25	Wait(Fa)	Wait(Mb)	Run
26	Wait(Fa)	Wait(Mb)	Ready
27	Wait(Fa)	Wait(Mb)	Wait(Mc)

【図１４】

(a)

タスク名 / 状態番号	タスクA	タスクB	タスクC
5	Run	Ready	Ready
6	Run	Ready	Wait(Mc)
8	Run	Wait(Mb)	Ready
20	Wait(Fa)	Run	Ready
21	Wait(Fa)	Run	Wait(Mc)
25	Wait(Fa)	Wait(Mb)	Run

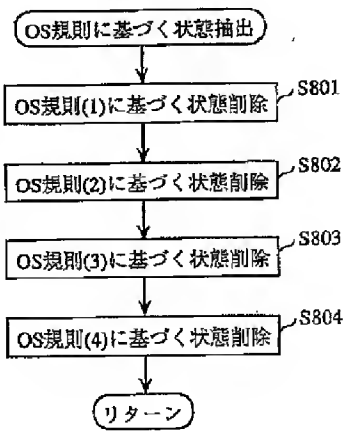
(b)

タスク名 / 状態番号	タスクA	タスクB	タスクC
1	Run	Ready	Ready
2	Run	Ready	Wait(Mc)
3	Run	Wait(Mb)	Ready
4	Wait(Fa)	Run	Ready
5	Wait(Fa)	Run	Wait(Mc)
6	Wait(Fa)	Wait(Mb)	Run

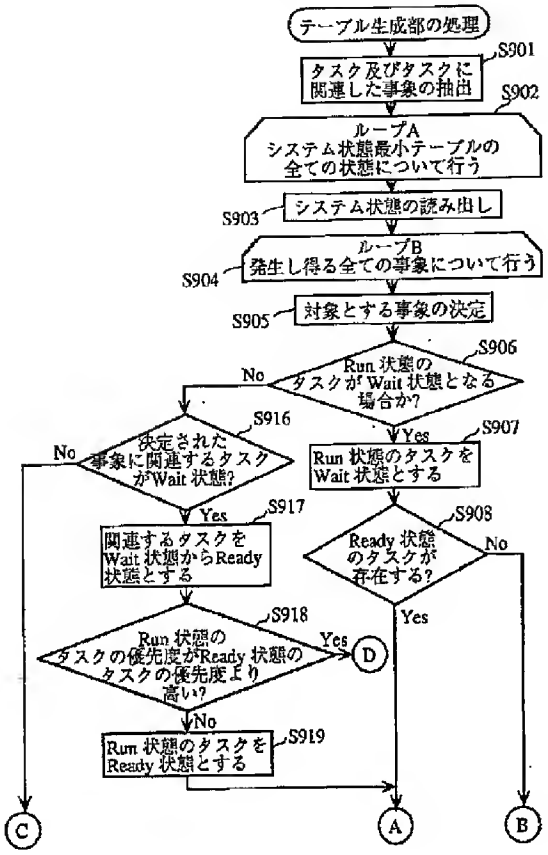
【図１５】

事象 / 状態番号	Wait状態への変化	フラグAの設定	メッセージBの送信	メッセージCの送信
1		—	—	—
2		—	—	1
3		—	1	—
4			—	—
5	—		—	4
6	—			—

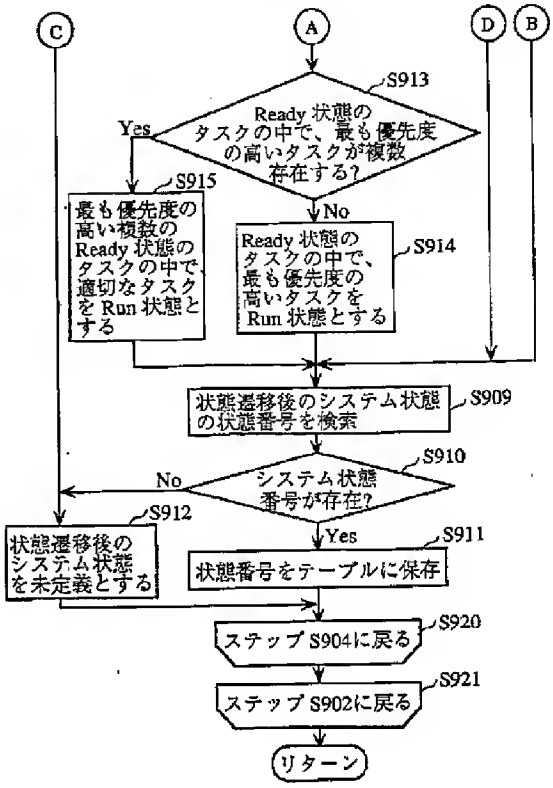
【図１７】



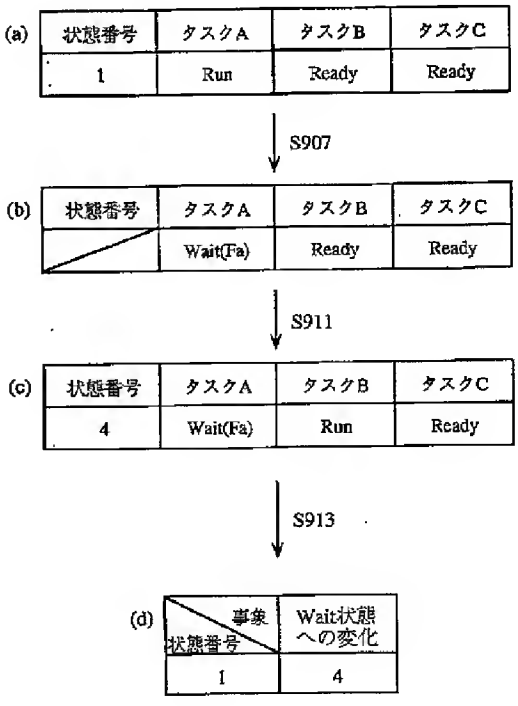
【図１８】


テーブル生成部の処理
S901
タスク及びタスクに関連した事象の抽出
S902
ループA
システム状態最小テーブルの全ての状態について行う
S903
システム状態の読み出し
ループB
S904
発生し得る全ての事象について行う
S905
対象とする事象の決定
S906
Run 状態のタスクが Wait 状態となる場合か?
No
Yes
S916
決定された事象に関連するタスクが Wait 状態?
No
Yes
S907
Run 状態のタスクを Wait 状態とする
S917
関連するタスクを Wait 状態から Ready 状態とする
S908
Ready 状態のタスクが存在する?
No
Yes
S918
Run 状態のタスクの優先度が Ready 状態のタスクの優先度より高い?
Yes
No
S919
Run 状態のタスクを Ready 状態とする
C
A
B
D

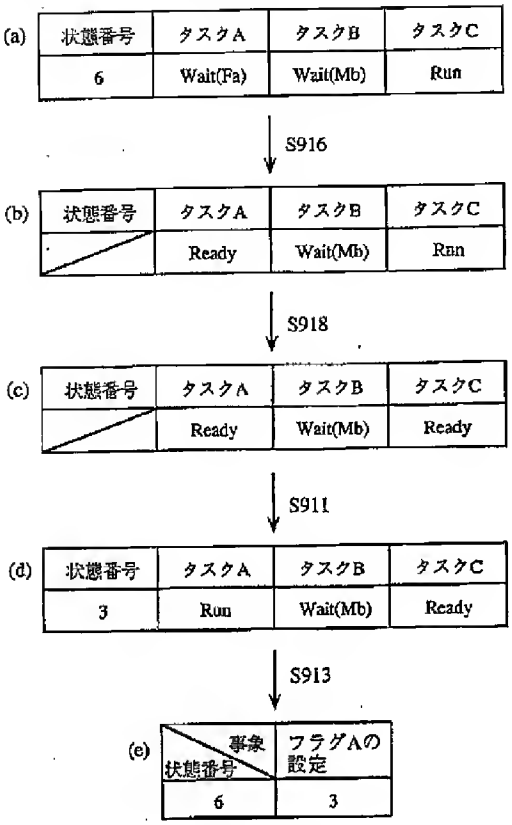
【図１９】


C
A
D
B
S913
Ready 状態のタスクの中で、最も優先度の高いタスクが複数存在する?
Yes
No
S915
最も優先度の高い複数の Ready 状態のタスクの中で適切なタスクを Run 状態とする
S914
Ready 状態のタスクの中で、最も優先度の高いタスクを Run 状態とする
S909
状態遷移後のシステム状態の状態番号を検索
S910
システム状態番号が存在?
No
Yes
S912
状態遷移後のシステム状態を未定義とする
S911
状態番号をテーブルに保存
S920
ステップ S904に戻る
S921
ステップ S902に戻る
リターン

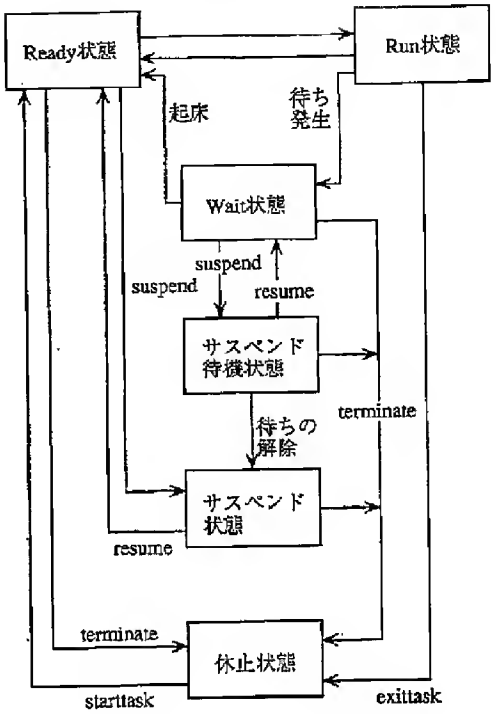
【図２０】


(a)
状態番号 | タスクA | タスクB | タスクC
1 | Run | Ready | Ready
S907
(b)
状態番号 | タスクA | タスクB | タスクC
Wait(Fa) | Ready | Ready
S911
(c)
状態番号 | タスクA | タスクB | タスクC
4 | Wait(Fa) | Run | Ready
S913
(d)
事象 / 状態番号 | Wait状態への変化
1 | 4

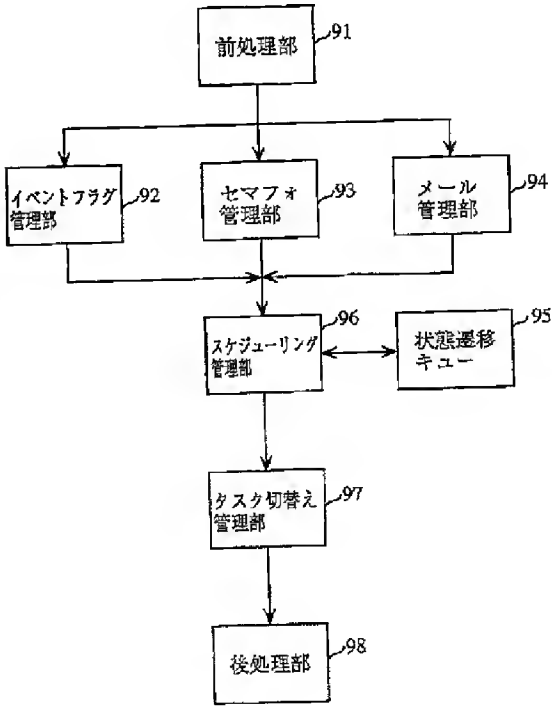
【図２１】


(a)
状態番号
タスクA
タスクB
タスクC
6
Wait(Fa)
Wait(Mb)
Run
S916
(b)
状態番号
タスクA
タスクB
タスクC
Ready
Wait(Mb)
Run
S918
(c)
状態番号
タスクA
タスクB
タスクC
Ready
Wait(Mb)
Ready
S911
(d)
状態番号
タスクA
タスクB
タスクC
3
Run
Wait(Mb)
Ready
S913
(e)
事象
状態番号
フラグAの設定
6
3

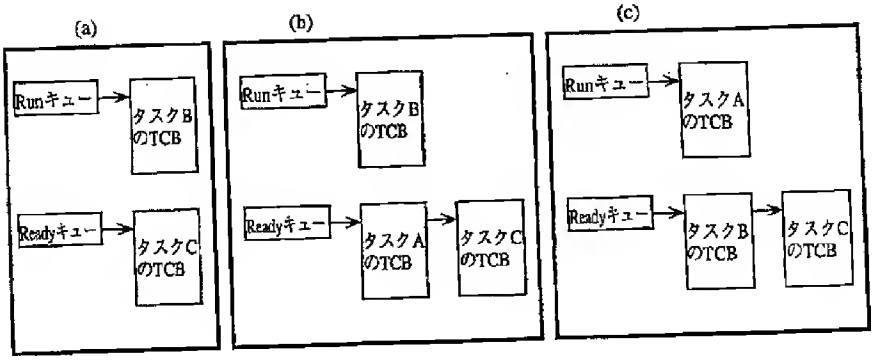
【図２２】


Ready状態
Run状態
起床
待ち発生
Wait状態
suspend
suspend
resume
サスペンド待機状態
terminate
待ちの解除
サスペンド状態
resume
terminate
休止状態
starttask
exittask

【図２３】


前処理部
91
イベントフラグ管理部
92
セマフォ管理部
93
メール管理部
94
スケジューリング管理部
96
状態遷移キュー
95
タスク切替え管理部
97
後処理部
98

【図２４】


(a)
Runキュー
タスクBのTCB
Readyキュー
タスクCのTCB
(b)
Runキュー
タスクBのTCB
Readyキュー
タスクAのTCB
タスクCのTCB
(c)
Runキュー
タスクAのTCB
Readyキュー
タスクBのTCB
タスクCのTCB